P.1677/70

maszyny matematyczne

zastosowania

w gospodarce

technice

i nauce

5
1970

SPIS TREŚCI

Str.

Przed VI Kongresem Techników Polskich — Tezy do dyskusji przedkongresowej 1

Jowita Koncewicz — „Języki do przetwarzania struktur listowych" 3

Andrzej W. Mostowski — „Zastosowania maszyn matematycznych w algebrze" 8

Władysław M. Turski — „Obecny kryzys w dziedzinie oprogramowania maszyn cyfrowych" 11

Janusz Ilczuk — „Cele i strategia zastosowań komputerów" 14

Stefan Andrejszyn — „Technika obliczeniowa w zastosowaniu do zagadnień normalizacji" 17

ZE ŚWIATA

„Rewolucja 1970" 22

WIADOMOŚCI PKAPI

„Drugie krajowe spotkanie użytkowników EMC ODRA" . . . . III str. okł.

PRZEGLĄD WYDAWNICTW

WYDAWNICTWA CZASOPISM TECHNICZNYCH NOT
Warszawa
Czackiego 3/5

KOLEGIUM REDAKCYJNE
Redaktor naczelny prof. dr Leon ŁUKASZEWICZ

Doc. dr hab. inż. Konrad FIAŁKOWSKI (zast. redaktora naczelnego), Władysław KLEPACZ, dr Antoni MAZURKIEWICZ, inż. Dorota PRAWDZIC (zast. redaktora naczelnego), dr inż. Andrzej TARGOWSKI

Sekretarz Redakcji mgr Wanda KAČER **Redaktor techniczny Bogdan DROZDOWSKI**

RADA PROGRAMOWA

Mgr inż. Jan Bursche, mgr inż. Henryk Chyrek, (wiceprzewodniczący) mgr inż. Ryszard Dąbrówka, mgr inż. Bolesław Gliksman, mgr inż. Józef Knysz, prof. dr Leon Łukaszewicz, mgr inż. Jan Matejak, prof. dr Tadeusz Peche (przewodniczący), mgr inż. Jerzy Trybulski (wiceprzewodniczący), dr Tadeusz Walczak, mgr Tadeusz Wasilewski, mgr Waldemar Wiśniewski (sekretarz), mgr Stefan Wojciechowski, dr inż. Henryk Woźniacki, mgr inż. Jan Zdzisław Żydowo

Redakcja: Warszawa, ul. Emilii Plater 20 m. 15, tel. 21-13-91. Zastępca redaktora naczelnego tel. 28-37-29

Zakład Kolportażu WCT NOT, Warszawa, ul. Mazowiecka 12

Zakł. Graf. „Tamka". Z. 2. Zam. 214. Papier powlekany V kl. 80 g. Obj. 3 ark. druk. Nakład 2800. K-43

Cena egzemplarza zł 8.— **Prenumerata roczna zł 96.—**

INDEKS 36707

P. 1877/70

maszyny matematyczne

zastosowania w gospodarce, technice i nauce

Nr 5

MIESIĘCZNIK

1970

ROK VI

Maj

Organ Pełnomocnika Rządu do Spraw Elektronicznej Techniki Obliczeniowej i Polskiego Komitetu Automatycznego Przetwarzania Informacji Naczelnej Organizacji Technicznej

PRZED VI KONGRESEM TECHNIKÓW POLSKICH

W marcu bieżącego roku zakończył się pierwszy etap prac przygotowawczych do VI Kongresu Techników Polskich, który odbędzie się w lutym 1971 roku w Poznaniu. W wyniku pierwszego etapu prac przygotowawczych Naczelna Organizacja Techniczna opublikowała tezy wszystkich sekcji głównych powołanych przez Główny Komitet Organizacyjny Kongresu.

Sekcje główne kierowały się w swej pracy naczelnym hasłem Kongresu: **„Technika w procesie intensyfikacji gospodarki"** i naczelnym zadaniem Kongresu, które ma polegać na wytyczeniu dla kadry naukowo-technicznej programu realizacji zadań gospodarczych na lata 1971—1975, nakreślonych uchwałami V Zjazdu PZPR i skonkretyzowanych przez II i IV Plenum KC PZPR.

Celem pierwszego etapu prac przygotowawczych do Kongresu było wyłonienie, dokonanie wyboru i sformułowanie tematyki Kongresu oraz powiązanie całego materiału w określoną całość. Nie była to łatwa praca ze względu na wielką ilość materiału, obejmującego wiele — często zazębiających się — dziedzin technicznych, jak również związane z nimi sprawy organizacji, ekonomii, planowania itd. Sekcje główne musiały równocześnie zadbać o zwięzłość sformułowań, o unikanie sformułowań sugerujących jednoznaczne stanowisko w poszczególnych problemach, o naświetlanie panującej sytuacji i przesłanek warunkujących możliwości rozwiązania dyskutowanych problemów.

Całość materiału opracowanego przez wszystkie sekcje główne obejmuje około 400 stron maszynopisu. Opublikowane tezy stanowią obecnie bazę do przeprowadzenia dyskusji przedkongresowej w kołach zakładowych, oddziałach stowarzyszeń naukowo-technicznych i w wojewódzkich komitetach organizacyjnych Kongresu. Mają one rolę kierunkującą, porządkującą i koordynującą.

W następnej fazie prac przedkongresowych oczekuje się szerokiej konfrontacji zawartych w nich sformułowań — z poglądami, informacjami i wnioskami powstałymi w wyniku dalszej dyskusji przedkongresowej w ramach organizacji społecznych. Oczekuje się, że dyskusja społeczna wyzwoli najszersze inicjatywy, przetrze drogi dla oddolnych koncepcji, wniosków i postulatów, które nie zawsze w sposób instytucjonalny mogą trafić tam, gdzie należy i być właściwie wykorzystane.

Oczekuje się również dużego zaktywizowania świata techniki, nie tylko w dyskusji, ale i w działaniu, tym bardziej że szeroki aktyw techniczny i gospodarczy właśnie w tym samym okresie zajmuje się pracami nad uruchomieniem i wykorzystaniem możliwości współczesnej nauki i techniki do przechodzenia z ekstensywnych form rozwoju gospodarki narodowej na formy intensywne, zgodnie z ostatnimi uchwałami Partii i Rządu.

Wypracowywanie najskuteczniejszych form i metod społecznego udziału w intensyfikowaniu i unowocześnianiu naszej gospodarki oraz w podwyższaniu jej efektywności wymaga połączenia wysiłków zarówno kadry naukowo-technicznej, jak i klasy robotniczej i ogółu pracujących. Dlatego Plenum CRZZ podjęło uchwałę o przystąpieniu instancji i organizacji związkowych oraz organów samorządu robotniczego do aktywnego udziału w przygotowaniu i przeprowadzeniu VI Kongresu Techników Polskich.

Problemy elektronicznej techniki obliczeniowej w zasadzie skoncentrowały się w tezach przygotowanych przez sekcje I i V, powołane przez Główny Komitet Organizacyjny Kongresu.

W niniejszym numerze „Maszyn Matematycznych" podajemy tę część tez Sekcji I („Technika i ekonomika — Problemy Zarządzania — Systemy przetwarzania informacji"), która jest najbliższa tematyki zawodowej naszych Czytelników.

Cytowany dalej rozdział tez Sekcji I nosi tytuł: „Informacja i systemy przetwarzania danych".

W najbliższym numerze naszego czasopisma podamy tekst odpowiedniej części tez przygotowanych przez Sekcję II („Elektronika, automatyka, telekomunikacja"), tj. rozdział pt.: „Elektroniczna technika obliczeniowa".

Zwracamy uwagę, że w materiałach przygotowanych przez inne sekcje główne również znajduje się wiele postulatów z zakresu stosowania w gospodarce maszyn matematycznych i systemów zautomatyzowanych.

Życzymy owocnej dyskusji przedkongresowej, która na pewno znajdzie odbicie i w naszym czasopiśmie!

Tezy do dyskusji przedkongresowej

SEKCJA I — Technika i ekonomika
Problemy zarządzania
Systemy przetwarzania informacji

Informacja i systemy przetwarzania danych

W latach 1971—1975 powinno nastąpić znaczne przyspieszenie ilościowego i jakościowego rozwoju stosowania elektronicznej techniki obliczeniowej w naszym kraju. Rozwój ten będzie warunkował między innymi realizację wytycznych w dziedzinie intensywnych metod gospodarowania i selektywnego rozwoju gospodarki. W tym celu należy przyjąć, że rozwój zastosowań elektronicznych maszyn cyfrowych oraz wszelkie działania z tym związane powinny należeć w perspektywie najbliższych lat do preferowanych dziedzin. Mimo znacznego wysiłku materialnego i organizacyjnego w latach 1971—75 nie będziemy w stanie nadrobić jeszcze dystansu, jaki dzieli nas w rozwoju zastosowań ETO nie tylko od wysoko rozwiniętych krajów kapitalistycznych, lecz i od CSRS i NRD. Według orientacyjnych danych — w r. 1975 w Polsce eksploatowanych będzie około 550 komputerów, podczas gdy w NRD — około 800, a w CSRS około 650.

Uwzględniając konieczność dokonania na tym odcinku istotnego postępu, zaleca się przedyskutowanie przez szeroki aktyw naukowy, techniczny i ekonomistów problematyki zawartej w niniejszych tezach.

1. Opracowanie programów pełnego wykorzystania istniejących ośrodków przetwarzania danych do celów zarządzania i koncepcji dalszego rozwoju sieci tych ośrodków.

W najbliższej perspektywie szczególne znaczenie będą miały systemy elektronicznego przetwarzania danych zakładowo-branżowe oraz systemy o zasięgu ogólnokrajowym.

2. Równoległe rozwijanie zastosowań maszyn matematycznych, analogowych, analogowo-cyfrowych oraz cyfrowych do sterowania procesami technologicznymi, zwłaszcza o charakterze ciągłym oraz do celów badawczo-projektowych.

3. Dokonanie właściwego wyboru kierunków zastosowań techniki cyfrowej oraz koncentracji prac i badań projektowych, wdrożeniowych w powiązaniu z rekonstrukcją w węzłowych dziedzinach organizacji i zarządzania. Wybór gałęzi, przedsiębiorstw oraz funkcji zarządzania jako węzłowych obiektów zastosowań wymaga opracowania wieloletnich programów wdrażania nowych technik zarządzania.

4. Dalszy szybki rozwój ogólnie dostępnej sieci ośrodków obliczeniowych, rozwijanych głównie w oparciu o powielanie rozwiązań wdrożonych już w ostatnich latach; wzmocnienie działalności projektowej ogólnodostępnej sieci ośrodków obliczeniowych, tematyka prac której powinna wynikać z programu wdrażania systemów elektronicznego przetwarzania danych — z uwzględnieniem priorytetów. Rozwijanie działalności usługowo-wdrożeniowej oraz projektowej dla określonych wytypowanych branż, kombinatów, przedsiębiorstw.

5. Stworzenie na bazie istniejącego potencjału takiej organizacji i systemu informacji naukowej, technicznej i ekonomicznej, aby zaistniała możliwość szybkiego uzyskiwania informacji dokumentacyjnej w postaci literatury naukowej, technicznej lub ekonomicznej.

6. Rozwijanie prac naukowo-badawczych w dziedzinie systemów informacyjnych jako bazy do dalszego planowego rozwoju elektronicznej techniki obliczeniowej w zarządzaniu i planowaniu gospodarczym. Zaliczyć tu można następujące kierunki:

- prace podstawowe w dziedzinie teorii systemów informacyjnych w powiązaniu z teorią podejmowania decyzji,
- prace w dziedzinie algorytmizacji procedur sterowania na poziomie procesów technologicznych oraz różnych szczebli zarządzania,
- prace w dziedzinie kodów (języków) dla różnych klas systemów informacyjnych z uwzględnieniem wymagań komunikacji „człowiek-maszyna-człowiek", w tym również prace w dziedzinie problemów klasyfikacji,
- prace nad teoretycznymi założeniami zintegrowanego systemu informacji ekonomicznej w powiązaniu z rozwiązaniami modelowymi w dziedzinie metod planowania i zarządzania,
- prace nad metodyką ogólną i szczegółową mierzenia sprawności i efektywności różnych klas systemów informacyjnych, pozwalające na formułowanie wytycznych dla wyboru optymalnych kierunków zastosowań, prowadzące do opracowania praktycznych metod,
- systematyczne prace nad metodyką zastosowań elektronicznej techniki obliczeniowej w projektowaniu, konstruowaniu, przygotowywaniu procesu technologicznego, produkcji i in.

7. Szersze wykorzystanie doświadczeń zagranicznych w pracach badawczo-wdrożeniowych i projektowaniu systemów automatycznego przetwarzania do celów zarządzania typowym przedsiębiorstwem, kombinatem przemysłowym i systemem branżowym.

8. Opracowanie koncepcji zautomatyzowanego systemu informacyjnego do potrzeb zespołu instytucji centralnych, przede wszystkim z punktu widzenia planowania krótko- i długookresowego oraz koncepcji stopniowej automatyzacji statystyki państwowej w powiązaniu z potrzebami kierownictwa. Nieodzowna jest praca w kierunku kształtowania założeń merytorycznych i techniczno-organizacyjnych „banku danych ewidencyjno-statystycznych", pozwalającego na szybki dostęp do nich wielu użytkownikom.

9. Uruchomienie prac nad wprowadzeniem w życie jednolitych w kraju systemów symbolizacji informacji w różnych grupach problemowych, znormalizowanie dokumentów pierwotnych.

10. Zorganizowanie jednostek (ew. biur projektowych) doradztwa organizacyjno-technicznego w problematyce praktycznego wdrażania elektronicznej techniki obliczeniowej u użytkowników. Poza tym celowe byłoby utworzenie wyspecjalizowanego przedsiębiorstwa, które w kooperacji z producentem krajowym maszyn i urządzeń lub importerami, biurami projektowymi i wyspecjalizowanym przedsiębiorstwem instalacyjno-montażowym — zajęłoby się całokształtem obsługi przyszłych użytkowników.

11. Powołanie wzorcowych ośrodków obliczeniowych, posiadających m. in. uprawnienia i odpowiednie środki do prowadzenia bibliotek programów o określonej specjalizacji. Uznanie za celowe stworzenie w aktualnych warunkach centralnej informacji o programach i systemach oraz zapewnienie warunków dla swobodnej wymiany programów. Rozważenie dalszego rozwoju działalności klubów i użytkowników maszyn matematycznych.

12. Ujednolicenie, (głównie z punktu widzenia software'u) parku maszynowego, stanowiącego niezbędny element kształtowania przyszłej krajowej sieci ośrodków obliczeniowych i efektywniejsze wykorzystanie kadry programistów i projektantów systemu.

13. Rozwinięcie prac nad stworzeniem krajowego systemu kształcenia i doskonalenia kadr dla dalszego rozwoju nowoczesnej techniki obliczeniowej oraz uwzględnienie problematyki szkolenia kadry kierowniczej różnych szczebli do różnych celów.

JOWITA KONCEWICZ
Instytut Maszyn Matematycznych
Warszawa

681.322.06

Języki do przetwarzania struktur listowych

Podano ogólną charakterystykę języków programowania, przeznaczonych do przetwarzania tekstów lub zbiorów elementów o zadanej strukturze. Powstały one podczas rozwiązywania zagadnień z takich dziedzin, jak praca konwersacyjna z maszyną, tłumaczenie tekstów, wyszukiwanie informacji, tworzenie bibliografii, rozpoznawanie postaci itd. Omówiono języki COMIT IPL, LISP i SLIP; opisano struktury danych, gospodarkę pamięcią, formalizm programowania oraz wspomniano o realizacji tych języków dla różnych maszyn cyfrowych.

1. WPROWADZENIE

Postępujący wciąż naprzód rozwój maszyn cyfrowych i towarzyszące mu rozszerzanie się możliwości ich zastosowań w coraz to nowych dziedzinach spowodowały pojawienie się nowych rodzajów danych, to jest wielkości, na których działają programy pisane dla tych maszyn. Dawniej operacje maszyn cyfrowych dotyczyły głównie danych liczbowych, mających postać bądź pojedynczych liczb całkowitych lub ułamkowych, bądź wektorów czy też wielowymiarowych macierzy liczbowych. Ostatnio coraz częściej danymi programów bywają zarówno informacje liczbowe, jak i tekstowe, powiązane zazwyczaj w bardzo skomplikowane struktury.

Nowe rodzaje danych powstają podczas rozwiązywania zagadnień z takich dziedzin, jak: praca konwersacyjna z maszyną, tłumaczenia tekstów, wyszukiwanie informacji, tworzenie bibliografii, symulowanie ludzkich procesów poznawczych, rozpoznawanie postaci, sztuczna inteligencja. We wszystkich tych, a także w wielu innych zagadnieniach występuje potrzeba działania na tekstach lub listach, tj. zbiorach elementów o zadanej strukturze, których elementami mogą być teksty pojedyncze lub listy list.

Równocześnie z pojawieniem się nowych rodzajów danych powstają nowe metody pracy przy stosowaniu maszyn cyfrowych. Tworzy się języki specjalne, przeznaczone do przetwarzania symboli i struktur listowych lub opracowuje techniki działania na tych danych środkami języków do zastosowań ogólnych, np. języków algorytmicznych takich, jak FORTRAN czy ALGOL, zrealizowanych już w przyjętym dla danej maszyny oprogramowaniu podstawowym. Celem tego artykułu jest ukazanie najważniejszych cech charakterystycznych języków do przetwarzania struktur listowych.

2. CHARAKTERYSTYKA OGÓLNA JĘZYKÓW DO PRZETWARZANIA STRUKTUR LISTOWYCH

Języki do przetwarzania struktur listowych różnią się między sobą dość znacznie zarówno pod względem ich składni, przyjętej reprezentacji struktur danych: zewnętrznej — w programie i wewnętrznej — w pamięci maszyny cyfrowej, metod przydzielania obszarów pamięci maszyny nowym strukturom oraz odzyskiwania pamięci zajętej przez dane już dłużej niepotrzebne, jak i sposobu ich realizacji w wielu różnych maszynach cyfrowych. Można jednakże dostrzec w nich wiele cech wspólnych, dzięki czemu możemy o nich mówić jako o odrębnej klasie języków wyspecjalizowanych.

Najistotniejszą cechą wspólną wszystkim językom do przetwarzania struktur listowych jest fakt, że służą do działania na danych tego samego szczególnego rodzaju. Nie są to wyłącznie dane liczbowe, lecz listy, teksty, struktury listowe, struktury pierścieniowe i drzewa dwójkowe. Przyjmiemy tutaj następujące definicje:

- Lista jest to dowolny ciąg elementów.
- Tekst jest to lista, której elementami nie mogą być listy.
- Struktura listowa lub drzewowa jest to lista, której elementami mogą być listy.
- Struktura pierścieniowa jest to lista, której element ostatni jest utożsamiony z pierwszym.
- Drzewo dwójkowe jest to rozszerzeniem pojęcia pary uporządkowanej, gdzie każdy element pary może być parą uporządkowaną.

Podstawowymi elementami tych struktur mogą być zarówno liczby, jak i napisy złożone z symboli liter, cyfr oraz symboli specjalnych konkretnej maszyny cyfrowej. Nie każdy z omawianych języków pozwala reprezentować w sposób jawny wszystkie wymienione wyżej rodzaje struktur danych, lecz na ogół jest możliwa reprezentacja pośrednia za pomocą struktur przewidzianych w formalizmie języka.

Inną cechą wspólną tym językom jest brak konieczności ustalenia z góry rezerwacji obszarów pamięci maszyny cyfrowej, gdyż przyznawane są one tworzonym strukturom danych w miarę powstawania tych struktur. Komórki pamięci przydzielane dynamicznie danym na ogół nie sąsiadują ze sobą. Każdy nowy element struktury dołącza się do niej za pomocą utworzenia ogniwa łączącego go z pozostałymi elementami lub dobudowania specjalnej „wskazówki" prowadzącej od danej struktury do elementu.

Języki do przetwarzania struktur listowych powinny umożliwiać przydzielanie miejsc pamięci strukturom tworzonym podczas wykonywania programu. W tym celu w pamięci maszyny, dla której zrealizowano jakiś język tego rodzaju, powinien być wydzielony obszar roboczy, dla programu zaś istnieć mechanizm pobierania wolnych komórek i zwracania niepotrzebnych. Wszystkie omawiane języki mają obszar pamięci roboczej dla programu, zorganizowany w postaci listy wolnych komórek pamięci, z której można je pobierać automatycznie, skracając tym samym tę listę. Komórki przydzielone strukturom, na których program przestał działać, a więc dalej niepotrzebne można zwracać liście wolnych komórek pamięci, lecz organizacja tego zwracania nie jest jednakowa w rozważanych językach.

Ponieważ obszar pamięci wewnętrznej przeznaczony na program i jego dane jest zazwyczaj ograniczony pojemnością pamięci rdzeniowej maszyny cyfrowej, języki do przetwarzania struktur listowych na ogół

przewidują możliwość dołączania do maszyny pamięci pomocniczych, zwłaszcza taśm oraz dysków magnetycznych. Powstają wówczas dodatkowe problemy związane z organizacją przetwarzania struktur danych w pamięciach dwupoziomowych, które też są rozmaicie rozwiązywane w różnych językach.

Ze względu na występującą często konieczność tworzenia lub przetwarzania struktur o bardzo złożonej budowie, wiele spośród języków do przetwarzania struktur listowych wprowadza możliwość definiowania podprogramów działających rekursywnie.

Podprogramem działającym r e k u r s y w n i e nazywamy taki, który może odwołać się do samego siebie przed zakończeniem swego działania. Podprogramy tego typu muszą mieć zapewnione przechowywanie w pamięci wyników częściowych i początkowych wartości argumentów, do czego świetnie nadają się pamięci stosowe, chroniące w sposób naturalny wartości otrzymane wcześniej, a udostępniające najświeższe.

P a m i ę c i ą s t o s o w ą lub w skrócie s t o s e m nazywamy taki ciąg elementów, w którym jest dostępny tylko pierwszy z nich, znajdujący się na tzw. „wierzchołku stosu". Nowy element można umieszczać tylko na wierzchołku stosu, a po usunięciu pierwszego elementu ze stosu, drugi element staje się pierwszym, a więc dostępnym elementem. Stąd jeszcze jedna cechą wspólną wielu omawianym tu językom jest korzystanie ze stosowej organizacji pamięci.

W następnych punktach przedstawimy cztery najszerzej znane na świecie języki do przetwarzania struktur listowych: COMIT, IPL, LISP i SLIP. Wydaja się one najlepszymi reprezentantami tej klasy języków, ponieważ zawieraja wszystkie cechy najbardziej charakterystyczne, a przy tym różnia się zasadniczo między soba, dzięki wyspecjalizowaniu ich do rozmaitych rodzajów zastosowań.

3. COMIT

Struktury danych. Elementami podstawowymi danych, na których działa program, są t e k s t y (ciągi symboli). Tekst zasadniczy, zawsze dostępny nazywa sie o b s z a r e m r o b o c z y m (ang. *workspace*) i złożony jest z elementów, czyli składowych rozdzielonych symbolami „+". Np. tekst perforowany w kartach:

A + SIMPLE + LINEAR + STRING + OF + CONSTITUENTS

lub prościej:

A SIMPLE LINEAR STRING OF CONSTITUENTS

jest przekształcany na tekst w języku COMIT, w którym każda litera lub odstęp sa traktowane jak składowe („—" reprezentuje symbol odstępu):

A+—+S+I+M+P+L+E+—+...+O+
+F+—+C+O+N+S+T+I+T+U+E+
+N+T+S

Programista może tworzyć jedną składową w obszarze roboczym z kilku innych w nim się znajdujacych, np. kilka pojedynczych liter scalać w jedno słowo. Nie wprowadza się ograniczeń co do zbioru symboli znajdujących się na klawiaturze danego urządzenia wejściowego maszyny cyfrowej, ani długości tworzonej składowej.

Składowe w obszarze roboczym moga mieć dowolna liczbę indeksów, którym można nadawać wartości. Liczby całkowite moga występować jedynie jako indeksy liczbowe, którym wówczas nie można przyporządkowywać wartości.

Obszar roboczy można traktować jako strukture listową o trzech poziomach: s k ł a d o w e, i n d e k s y i w a r t o ś c i. Nie można natomiast reprezentować bezpośrednio struktury o niezdefiniowanej liczbie poziomów. Przyjmuje się, że składowe w obszarze roboczym są uporządkowane, natomiast na indeksach i wartościach można działać, jak na zbiorach nie uporządkowanych.

Gospodarka pamięcią. Programista nie ma dostępu do aktualnej struktury listowej znajdującej się w pamięci. Stąd likwidowaniem niepotrzebnych informacji zajmuje się system COMIT. Interpreter języka COMIT prowadzi wykaz wszystkich zajętych komórek pamięci i z chwilą, gdy któraś nie jest dalej potrzebna, automatycznie umieszcza ją na liście wolnych komórek pamięci. Teksty można przesyłać z obszaru roboczego do pamięci pomocniczej, będącej pewną odmianą pamięci stosowej, w której można umieszczać elementy na dowolnym jej końcu, ale pobierać jedynie z początku. Można też przesyłać dowolne teksty między obszarem roboczym w pamięci wewnętrznej a pamięcią na taśmie lub dysku magnetycznym.

Formalizm programowania. Podstawową jednostką programu w języku COMIT jest r e g u ł a (ang. *rule*). Reguły definiują przekształcenia poszczególnych tekstów, podając żądaną postać wejściową i wyjściową danych. Nie można natomiast określać przekształceń tekstów za pomocą procedur. Podczas wykonywania danej reguły przeszukuje się obszar roboczy, dopóki nie natrafi się na odpowiadający tej regule podtekst, a w przypadku nieodnalezienia zadanego, pozostawia się obszar roboczy nie zmieniony i przechodzi do wykonywania następnej reguły. Reguły mogą określać przekształcenia tekstów w obszarze roboczym, ich indeksów lub wartości indeksów, a także powodować przesyłanie napisów z obszaru roboczego do pamięci pomocniczych i na odwrót. W przypadku zastosowania pewnego (dość zresztą niewygodnego) systemu łączników, można z grup reguł korzystać podobnie, jak z podprogramów.

W języku COMIT można opisać procesy rekursywne, aczkolwiek formalizm nie jest wygodny do określenia niezbędnych przy tego rodzaju działaniach wymagań przechowywania wyników częściowych czy też powrotów do podprogramów. Samomodyfikację programów można otrzymać w bardzo ograniczonym stopniu, polega ona na wybieraniu odpowiednich reguł za pomoca obliczania ich nazw znajdujących się w obszarze roboczym.

Arytmetyka jest również bardzo uboga, ponieważ dotyczy tylko liczb całkowitych umieszczanych jako indeksy w obszarze roboczym.

Natomiast szczególnie użyteczna cecha tego jezyka jest tzw. r e g u ł a l i s t y (ang. *list rule*), umożliwiająca szybkie przeszukiwanie słownika.

Realizacja. COMIT zrealizowano dla maszyn cyfrowych IBM 7090/94 oraz IBM 7040/44. W obu przypadkach translator tłumaczy reguły języka na reprezentacje pośrednia w języku maszyny, a interpreter wykonuje program.

4. IPL

Struktury danych. Elementami podstawowymi danych w języku IPL są s t r u k t u r y l i s t o w e definiowane za pomocą list symboli. Listę danych określa n a z w a l i s t y (symbol języka IPL) i ciag symboli tworzących konkretną listę. Symbole mogą być r e g i o n a l n e lub l o k a l n e.

Symbole regionalne są to nazwy list lub komórek pamięci. Symboli lokalnych używa się do nazywania podlist struktur listowych. Symbolami języka moga być również nazwy s ł ó w d a n y c h (ang. *data terme*), to jest komórek zawierających dane liczbowe lub literowo-cyfrowe, na których można działać za pomocą specjalnych procedur języka wywołanych poprzez etykiety.

W reprezentacji wewnętrznej struktur języka IPL każda komórka pamięci maszyny składa sie z czterech segmentów, z których pierwsze dwa sa polami o długości adresu, zwanymi SYMB i LINK, a dwa pozostałe — polami trzybitowymi, zwanymi *P* i *Q*.

Pola *P* i *Q* wskazują, czy komórka zawiera element listy, czy też słowo danych. W pierwszym przypadku

w polu SYMB znajduje się symbol języka IPL, w polu LINK zaś adres komórki, w której znajduje się następny symbol listy. Przyjmuje się, że ostatnia komórka listy zawiera liczbę 0 w polu LINK. Programista nie interesuje się konkretnymi adresami umieszczonymi w polu LINK. W przypadku danych pola *P* i *Q* wskazują rodzaj danych, które znajdują się w polach SYMB i LINK.

Gospodarka pamięcią. Programista sam określa, które struktury mają być przetwarzane, które zaś zlikwidowane. Każda podlista jest lokalna w danej strukturze. Pozostawienie w pamięci struktur, na których program nie będzie już działał, powoduje niepożądane uszczuplenie miejsca w pamięci maszyny.

W języku IPL istnieje kilka rozkazów przeznaczonych do wykonywania operacji na pamięciach stosowych. Natomiast w reprezentacji wewnętrznej nie czyni się rozróżnienia pomiędzy stosami a listami, dzięki czemu każdą strukturę można traktować dwoiście. Przyjmuje się jednak czasem, że pewne symbole są nazwami komórek pamięci stosowej, a inne nazwami list. Dane, a także części programu mogą być przechowywane na taśmie magnetycznej i w razie potrzeby wprowadzane automatycznie do pamięci operacyjnej.

Formalizm programowania. Procedura w języku IPL jest ciągiem rozkazów wykonywanych kolejno z wyjątkiem punktów skokowych, w których przewiduje się możliwość wyboru jednego z dwóch rozgałęzień na podstawie zawartości jakiejś badanej komórki. Rozkazy mogą określać operacje podstawowe, zmieniające stan pamięci stosowej lub listy danych, albo mogą powodować przejście do wykonywania innej procedury, po której wykonaniu sterowanie wraca do następnego rozkazu procedury wywołującej. Postępowanie takie może przebiegać rekursywnie. Interpreter języka IPL prowadzi w pamięci stosowej ślad adresów rozkazów wykonywanych aktualnie i kończy działanie z chwilą zakończenia wykonywania procedury znajdującej się na najwyższym poziomie w hierarchii wywoływania procedur. Programista jest odpowiedzialny za dostarczanie procedurom argumentów umieszczanych w pamięci stosowej, zwanej komórką połączeń oraz musi dbać o to, aby nie doszło do konfliktów podczas korzystania z wyników częściowych.

Modyfikacja programu jest możliwa dzięki temu, że nazwy procedur są symbolami języka IPL, system zaś zawiera rozkazy powodujące wykonanie procedur, których nazwy dostarcza się jako dane programu. Operacje podstawowe języka IPL pozwalają podczas wykonywania programu konstruować dowolne listy o budowie zażądanej przez jakąś procedurę i następnie wykonywać tę procedurę dla utworzonej już listy. Istniejące procedury można przekształcać za pomocą programu tak, jak listy danych.

Możliwości wykonywania działań arytmetycznych w systemie IPL są dość ograniczone ze względu na konieczność przechowywania w komórkach pamięci liczb w postaci zgodnej ze specjalnymi wzorcami oraz ze względu na duże zmniejszenie szybkości wykonywania programu wynikające z interpretacji tych wzorców przez rozkazy arytmetyczne.

W języku IPL istnieje bardzo cenna możliwość działania na listach-opisach (ang. *description-list).* Każdej liście można przyporządkować listę-opis składającą się z par elementów. Operacje języka IPL pozwalają dołączać do tych list pary elementów oraz wybierać drugi element pary wówczas, gdy podany jest pierwszy element i nazwa listy głównej.

Ponadto można pisać programy w języku IPL generujące dowolne ciągi symboli jako dane wejściowe dla innego programu. Można napisać np. program-generator, który pozwala na optymalne rozplanowanie wykorzystania miejsc pamięci maszyny.

Realizacja. Język IPL zrealizowano jako system interpretacyjny dla maszyn: IBM 709/90, CDC 1604, UNIVAC 1105, BENDIX G-20, PHILCO 2000, AN/FSQ 32 i IBM 650. Dla maszyny IBM 709/90 napisano eksperymentalnie translator kompilacyjny, lecz okazało się, że program wynikowy zajmuje dużo więcej miejsca i działa nieznacznie szybciej od identycznego programu wykonywanego interpretacyjnie.

5. LISP

Struktury danych. Elementami podstawowymi danych w języku LISP 1.5, o którym będzie mowa, nazywanymi symbolami atomowymi, czyli w skrócie atomami (ang. *atom),* są ciągi co najwyżej trzydziestu symboli liter i (lub) cyfr. Symbole atomowe można łączyć w drzewa dwójkowe lub struktury listowe. Drzewa dwójkowe można przedstawiać bezpośrednio za pomocą notacji z kropką (ang. *dot notation),* przy czym napis: (A.B) jest uporządkowaną parą atomów A i B. Na miejscu symboli A i B można umieszczać dowolne pary symboli z kropką określając w ten sposób drzewo dwójkowe, np.

((E.E).(H.(J.K)))

Za pomocą notacji listowej (ang. *list notation)* można stosując przecinki przedstawiać listy atomów lub listy list, np.:

(PEOPLE, (BOYS, (TOM, DICK)), (GIRLS, (JANE)), (OTHERS,()))

Podlisty umieszcza się w liście w sposób jawny. Lista w powyższym przykładzie składa się z czterech elementów, z których drugi, trzeci i czwarty są listami; czwarta lista ma dwa elementy: pierwszym elementem jest atom OTHERS, drugim — lista pusta.

Obie notacje można stosować zamiennie i równocześnie, przyjmując, że lista
S1, S2, ..., Sn

jest równoważna parze w notacji z kropką

(S1.(S2...(Sn.NIL)...))

przy czym atom NIL spełnia rolę symbolu terminatora oznaczającego koniec listy i odpowiada liście pustej.

W reprezentacji wewnętrznej nie odróżnia się drzew dwójkowych w notacji z kropką od struktury listowej i przedstawia się je jako drzewa dwójkowe komórek pamięci. Podobnie, jak w języku IPL, każda komórka struktury danych zawiera dwa pola adresu służące do określania podwyrażeń. Symbole atomowe gromadzi się na specjalnej liście własności (ang. *property list)* zawierającej ich reprezentację zewnętrzną i pewne informacje dodatkowe. Atomy rozpoczynające się od cyfry są traktowane odrębnie jako liczby i, w celu odróżnienia, opatrywane w specjalne znaczniki.

Gospodarka pamięcią. Programista nie musi śledzić, które komórki pamięci są aktualnie zajęte podczas wykonywania programu, co zresztą może okazać się niewykonalne ze względu na bardzo skomplikowane powiązania między nimi. Nie czyni tego również system; z chwilą, gdy brakuje miejsca w pamięci na tworzone struktury danych, system dokonuje operacji oczyszczania pamięci (ang. *garbage colection).* Pozostawia wówczas nienaruszone te struktury, do których program odwołuje się aktualnie, a udostępnia jako wolne wszystkie inne, zajęte poprzednio komórki pamięci. Programista nie działa również na pamięci stosowej, z której korzysta system, np. w celu wykonywania wywołań rekursywnych, wprowadzając dość dowolny podział pamięci na obszar przeznaczony na struktury listowe i obszar pamięci stosowej.

Formalizm programowania. Programy w języku LISP 1.5 nie są ciągami rozkazów i opisów postaci danych, lecz są zbudowane z wyrażeń funkcyjnych, w których jako argumenty wywoływanych funkcji można podstawiać wartości innych funkcji. System LISP 1.5 zawiera podstawowy zbiór funkcji języka. Na ich podstawie programista może definiować nowe funkcje, stosując wyrażenia warunkowe dowolnej długości typu:

if p_1 *then* e_1 *else if* p_2 *then* e_2... *if pi then ei else*

gdzie p_i są predykatami, e_i $(i = 1,2,...)$ — dowolnymi wyrażeniami w języku LISP 1.5. Wartością wyrażenia warunkowego jest wartość wyrażenia e_j, jeżeli p_j jest pierwszym, sprawdzając od lewej, predykatem przyjmującym wartość p r a w d a.

Formalizm języka LISP 1.5 jest szczególnie wygodny dla rekursywnego definiowania funkcji. Podczas obliczania wartości funkcji rekursywnych przechowuje się wszystkie wyniki częściowe. Programista nie musi opisywać dokładnie struktur danych, na których jego program będzie działał, które wyniki częściowe można usunąć, a które przechować.

W definicji funkcji rekursywnej nie ogranicza się z góry liczby odwołań (tzw. zagłębień rekursji) ani liczby kolejnych podstawień wartości jakichś funkcji na miejsca argumentów dowolnej funkcji. Definicje funkcji są reprezentowane jako struktury listowe, a więc strukturalnie równoważne danym. Podczas obliczania wartości funkcji, system interpretuje jej argumenty jako reprezentację listy dla funkcji. Dzięki tym cechom systemu, programy mogą być w pełni samomodyfikujące się, ponieważ podczas wykonywania programu można zmieniać definicje funkcji, czy też tworzyć nowe funkcje — podstawiając definicje innych funkcji na miejsca argumentów funkcji określonych wcześniej.

System LISP 1.5 zawiera pełny zbiór operacji arytmetycznych. Operacje te jednak wykonują się dość powoli ze względu na konieczność wyszukiwania liczb na liście własności umieszczonych tam jako symbole atomowe.

Język LISP 1.5 pozwala również definiować funkcjonały lub funkcje o argumentach zawierających definicje funkcji. Można np. określić funkcję dwuargumentową map $[x, f]$, której wartością jest lista wartości funkcji f dla każdego elementu listy x. Można wreszcie definiować funkcje jako listy kroków programu, wykonywanego podobnie, jak w językach algorytmicznych, np. w FORTRAN-ie.

Realizacja. Język LISP 1.5 zrealizowano m. in. dla maszyn: IBM 1620, IBM 709/90, AN/FSQ-32, PDP-1 i AFCRL M460. System dla IBM 709/90 składa się z monitora, interpretera i kompilatora. Kompilator służy do tłumaczenia definicji funkcji ze struktury listowej na podprogramy w języku maszyny (można z niego nie korzystać lub nawet usunąć go z systemu). Monitor steruje rozmieszczaniem danych w pamięci i rozpoczyna wykonywanie działań niezależnych.

6. SLIP

Struktury danych. Elementami podstawowymi danych w języku SLIP są liczby i ciągi symboli literowo-cyfrowych o długości zmieniającej się zależnie od długości słowa maszyny cyfrowej. Z tych elementów można tworzyć struktury listowe, na których działa program. SLIP nie jest systemem niezależnym, lecz zbiorem podprogramów w systemie FORTRAN. Dzięki temu można wykonywać działania na liczbach zgodnie z zasadami przyjętymi w FORTRAN-ie. Nie ustala się reprezentacji zewnętrznej danych, które czyta się i dołącza do list na podstawie określeń dostarczanych przez programistę w instrukcjach wejścia-wyjścia. W reprezentacji wewnętrznej struktur listowych pozycję pojedynczą listy tworzy para komórek pamięci związanych ze sobą.

Pierwsza z nich zawiera dwa pola o długości adresu i jedno pole dwubitowe charakteryzujące rodzaj pozycji. Pola o długości adresu, zwane lewym i prawym o g n i w e m wskazują odpowiednio pozycje poprzednią i następną na danej liście. Druga komórka pary zawiera aktualną pozycję listy, którą może być zarówno całe słowo danych, jak i wskaźnik podlisty. Każda lista ma n a g ł ó w e k (ang. *header*) składający się z pary komórek i służący do wykonywania wszelkich odwołań do danej listy. Ogniwa nagłówka wskazują pierwszy i ostatni element listy.

Gospodarka pamięcią. Programista sam zajmuje się likwidowaniem list już niepotrzebnych. System pozwala wielokrotnie korzystać z danej listy i dla usprawnienia likwidacji list prowadzi w nagłówku każdej listy l i c z n i k o d w o ł a ń, określający liczbę odwołań do tej listy ze struktur znajdujących się aktualnie w pamięci maszyny. Przy każdym zlikwidowaniu jakiejś komórki danej listy zmniejsza się zawartość licznika odwołań tej listy. W chwili, gdy jest ona równa zeru wszystkie komórki tej listy są zwrócone liście wolnych komórek pamięci. Nie wprowadza się rozróżnienia między listami a podlistami i system nie likwiduje podlisty struktury listowej aktualnie likwidowanej dopóty, dopóki komórki tej listy są rzeczywiście jeszcze potrzebne. Nie czyni się też rozróżnienia między pamięcią stosową a zwykłymi listami.

Można odwoływać się do komórek wewnątrz danej listy oraz dopisywać informacje do obu końców listy. W celu przesyłania danych między pamięcią wewnętrzną a pamięciami pomocniczymi można posłużyć się instrukcjami wejścia/wyjścia języka FORTRAN.

Formalizm programowania. Ponieważ system SLIP jest zbiorem podprogramów, które można dołączyć do każdego systemu FORTRAN, składnia jego języka jest identyczna ze składnią języka FORTRAN. Podprogramy SLIP-u są zbudowane z ciągów rozkazów dokonujących przekształceń struktur danych. Do sterowania przebiegu wykonywania podprogramów służą instrukcje FORTRAN-u. IF i GO TO. Dzięki własnościom języka FORTRAN, SLIP pozwala tworzyć funkcje przetwarzające listy o możliwościach zbliżonych do funkcji uzyskiwanych w języku LISP 1.5. Dzięki specjalnym funkcjom v i s i t (odwiedź) i a s s i g n (przypisz) można określać podprogramy działające rekursywnie, jednak programista musi określać w sposób jawny, które argumenty należy przesłać i które wyniki pośrednie zachować. Programy mogą być samomodyfikujące się tylko w bardzo ograniczonym stopniu: można jedynie za pomocą procedur działających na symbolach określać porządek wykonywania fragmentów innych zaprogramowanych procedur. Natomiast możliwości wykonywania operacji arytmetycznych są w tym języku największe spośród omawianych tutaj języków do przetwarzania struktur listowych, ponieważ SLIP korzysta w pełni z arytmetyki języka FORTRAN. Ponadto SLIP zawiera operacje (podprogramy) działające na pojedynczych symbolach i bitach, a więc pozwala również działać na częściach słów maszyny.

Realizacja. SLIP można bardzo łatwo zrealizować dla dowolnej maszyny, mającej translator języka FORTRAN lub języka podobnego typu. Większość podprogramów można napisać w FORTRANIE, a tylko niektóre w języku symbolicznym danej maszyny. Zwykle przyjmuje się organizację systemu pozwalającą dołączyć za pośrednictwem monitora procedury SLIP-u i przy tym tylko te, które są aktualnie potrzebne podczas wykonywania danego programu. SLIP zrealizowano między innymi dla maszyn: IBM 709/90, AN/FSQ-32, PDP-1 i AFCRL M460.

7. UWAGI KOŃCOWE

Należy tutaj podkreślić, że przedstawione omówienie języków do przetwarzania struktur listowych jest bardzo pobieżne i oparte na literaturze prezentującej osiągnięcia w tej dziedzinie uzyskane do roku 1968. W owym czasie było już co najmniej kilkanaście innych tego rodzaju języków, dość szeroko stosowanych na świecie; można spodziewać się, że w chwili obecnej jest ich znacznie więcej.

Pierwszym językiem przeznaczonym do przetwarzania napisów opracowanym w Polsce był LOGOL, pozwalający symulować maszynę Turinga lub dowolny automat skończony, zrealizowany w roku 1964 w Instytucie Maszyn Matematycznych dla maszyny ZAM-2.

W kilka lat później powstały dwie realizacje języka LISP 1.5. na Uniwersytecie Warszawskim dla maszy-

ny GIER. Pierwsza z nich — to interpreter LISP-u działający jako system niezależny, druga — jest zanurzeniem tego języka w ALGOL-u przez dołączenie do systemu GIER ALGOL-4 procedur interpretujących funkcje podstawowe języka LISP 1.5.

W latach 1965—66 powstał w Instytucie Maszyn Matematycznych język do przetwarzania symboli EOL, którego istnieje kilka wersji, zrealizowanych zarówno w Polsce dla maszyny ZAM-41, jak i w Stanach Zjednoczonych AP dla maszyn IBM 7094 i IBM 360. Bliższe szczegóły dotyczące tego języka zostały opublikowane m.in. w zeszycie nr 5/1969 „Maszyn Matematycznych".

Jako uzupełnienie niniejszego artykułu podajemy dość obszerną bibliografię ułożoną w porządku chronologicznym, na ogół dostępną w Polsce.

BIBLIOGRAFIA

[1] Mc Carthy J.: Recursive Functions of Symbolic Expressions and Their Computation by Machine; Comm. ACM 3, 4 (Apr. 1960)

[2] Perlis A. J. and Thornton C.: Symbol Manipulation by Threaded Lists; Comm. ACM 3, 4 (Apr. 1960)

[3] Newell A., Tonge F. M.: An Introduction to Information Processing Language V, Comm. ACM 3, 4 (Apr. 1960)

[4] Green B. F.: Computer Languages for Symbol Manipulation; IRE Trans. HFE 2 (March 1961)

[5] Yngve V. H.: COMIT as an IR Language; Comm. ACM 5, 1 (Jan. 1962)

[6] Mc Carthy J. et al.: LISP 1.5 Programmer's Manual; MIT Press, Cambridge, (Mass. 1962)

[7] Newell A.: Documentation of IPL-V; Comm. ACM6, no 3 (March 1963)

[8] Weizenbaum J.: Symetric List Processor; Comm. ACM 6, no 9 (September 1963)

[9] Baecker H. D.: Mapped List Structures; Comm. ACM 6, 8 (Aug. 1963)

[10] Newell A. (ed.): Information Processing Language-V Manual; Prentice Hall, Englewood Cliffs N. J., 2 nd edition (1963)

[11] Brown W. S. et al.: The ALPAK System for Nonnumerical Algebra on a Digital Computer; Bell System Tech. J. 42, pp. 2081-2119 (1963); 43, pp. 785—804 (1964); 44, pp. 1547—1562 (1964)

[12] Bobrow D. G.; and B. Raphael: A Comparison of List-Processing Computer Language; Comm. ACM 7. No 4 (April 1964)

[13] Wilkes M. V.: Lists and Why They Are Useful; Proc. 19th Nat. Conf. ACM (August 1964)

[14] Farber D. J. et al.: SNOBOL, a String-Manipulation Language J. ACM 11, No 1 (January 1964)

[15] Roberts L. G.: Graphical Communication and Control Languages, Second Conf. on Information System Science, Hot. Springs, Va. (1964)

[16] Bobrow D. G. and J. Weizenbaum: List Processing and Extension of Language Facility by Embedding; IEEE Trans. EC 13, no 4 (August 1964)

[17] Christensen C.: AMBIT: A Programming Language for Algebraic Symbol Manipulation; CA/64/4A, Computer Associates, Inc., Wakefield Mass. (October 1964)

[18] Reynolds J.: COGENT Programming Manual; ANL-7022, Argonne National Laboratory (March 1965)

[19] Haines E. C.: The Treet List-Processing Language; SR-133, Mitre Corp., Bedford, Mass. (April 1965)

[20] Reynolds J.: An Introduction to the COGENT Programming System; Proc. 20 th Nat. Conf. ACM (August 1965)

[21] Brown W. S.: A Language and System for Symbolic Algebra on a Digital Computer; Proc. IBM Scientific Computing Symposion on Computer/Aided Experimentation (October 1965)

[22] Strachey C.: General-Purpose Macrogenerator; Computer J. (October 1965)

[23] Cohen K., and J. H. Wegstein: AXLE: an Axiomatic Language for Strings Transformations; Comm. ACM 8, No 11 (November 1965)

[24] Sammet J. E.: Formula Manipulation by Computer; Tr 00.1363, IBM Systems development Division, Poughkeepsie, New York (November 1965)

[25] Sakoda J. M.: DYSTAL Manual Sociology Computer Laboratory, Brown University, Providence, E. I. (1965)

[26] FORMAC Manual; IBM Corp., Program Information Dept., 40 SAW Mill River Road, Hawthorne, N. Y.

[27] Berkeley E. C. and D. G. Bobrow (Eds.): The Programming Language LISP: Its Operation and Applications; MIT Press, Cambridge, Mass. (1966)

[28] LISP 2, Document Series TM-2710, System Development Corp., 2500 Colorado Avenue, Santa Monica, Calif. (1966)

[29] Mooers C.: TRAC, a Procedure Describing Language for the Reactive Typewriter; Comm. ACM 9, No 3 (March 1966)

[30] Teitelman W.: FLIP-a Format List Processor; Memo MAC-M-263, MIT Project Mac, Cambridge, Mass. (1966)

[31] Farber D. J. et al.: The SNOBOL 3 Programming Language; Bell System Tech. J. (July-August 1966)

[32] Guzman A. and H. Mc Intosh: CONVERT; Comm. ACM 9, No 8 (August 1966)

[33] Caracciolo A. et al.: PANON-1B, a Programming Language for Symbol Manipulation; Comm. ACM 9, No 8 (August 1966)

[34] Christensen G.: On the Implementation of AMBIT, a Language for Symbol Manipulation; Comm. ACM 9, No 8 (August 1966)

[35] Raphael B.: Aspects and Applications of Symbol Manipulation; Proc. 21st Nat. Conf. ACM (August 1966)

[36] Cohen J.: A Use of Fast and Slow Memories in List-Processing Languages; Comm. ACM 10, 2 (Febr. 1967)

[37] Bobrow D. G. and Murphy D. L.: Structure of a LISP System Using Two-Level Storage; Comm. ACM 10,3 (Mar. 1967)

[38] Lawson H. W. Jr.: PL/1 List Processing; Comm. ACM 10, 6 (June 1967)

[39] Schorr H. and Waite W. M.: An Efficient Machine-Independement Procedure for Garbage Collection in Various List Structures; Comm. ACM 10, 8 (Aug. 1967)

[40] Smith D. K.: An Introduction to the List-Processing Language SLIP; Programming System and Languages, ed. S. Rosen, McGraw-Hill, 1967

[41] Bobrow D. G.: Storage Management in LISP; Symbol Manipulation Languages and Techniques, North-Holland, Amsterdam 1968

[42] Benzécri J. P.: Symbol Manipulation and General Algebraic Structures; Symbol Manipulation Languages and Techniques, North-Holland, Amsterdam 1968

[43] Hoare C.A.R.: Data Structures in Two-Level Store; IFIP Congress-68. Edinburgh, North-Holland, Amsterdam

[44] Engeli M. E.: A Language and List Structures for an Algebraic Manipulation System; Symbol Manipulation Languages and Techniques. North-Holland, Amsterdam 1968

[45] Mysior J. and Waligórski S.: LOGOL — a String Manipulation Language; Symbol Manipulation Languages and Techniques, North-Holland, Amsterdam 1968

[46] Łukaszewicz L.: EOL język do przetwarzania symboli, „Maszyny Matematyczne" nr 5/1969

[47] Waligórski S.: Przetwarzanie napisów — wyniki prac prowadzonych na Uniwersytecie Warszawskim. Problemy naukowe maszyn matematycznych. Materiały konferencji, Zakopane 1968 (w druku).

CZYTELNICTWO PRASY TECHNICZNEJ —

DROGĄ DO POGŁĘBIANIA WIEDZY

PUBLIKOWANIE WŁASNYCH FACHOWYCH ARTYKUŁÓW —

ŚWIADECTWEM DOJRZAŁOŚCI ZAWODOWEJ

Andrzej Włodzimierz Mostowski
Uniwersytet Warszawski

681.322.004:512:519.41/.47

Zastosowania maszyn matematycznych w algebrze

Omówiono kierunek zastosowań komputerów w algebrze związany z badaniem grup przedstawionych przez generatory i relacje. Niektóre metody są wykorzystywane praktycznie w dziedzinie fizyki i krystalografii.

Najbardziej typowe i najszerzej rozpowszechnione zastosowania maszyn matematycznych w algebrze polegają na pomocy maszyny przy badaniu własności pewnych tworów algebraicznych przez bezpośrednie wyznaczanie struktury tych tworów bądź przez bezpośrednie przekształcanie przez maszynę pewnych wyrażeń, bądź też przez stawianie hipotez dotyczących pewnych faktów i sprawdzanie ich przez maszynę.

Rachunki w algebrze dalece odbiegają od rachunków na liczbach, wykonuje się je na symbolach, tzn. literach lub zmiennych, przy tym prawa algebraiczne są dla różnych algebr zupełnie odmienne, gdyż opierają się na odmiennych własnościach działań. Wielekroć ręczne wykonywanie żądanych przekształceń jest żmudne i wymaga dużo wprawy. Mimo to wykonuje się je ręcznie w szerokim zakresie.

Można jednak te rachunki wykonywać na maszynach cyfrowych za pomocą języków specjalnie przystosowanych do takich operacji. Warto wymienić tu przykładowo język FORMAC stanowiący odmianę FORTRAN-u IV, ALPAC, LISP oraz ALGOL-68 (jeszcze nie opracowany ostatecznie).

Rachunki formalne wykonywane przez maszynę są jeszcze praktycznie mało wydajne i czasochłonne. Podlegają ponadto różnym ograniczeniom związanym z parametrami maszyny. Z trudnościami występującymi w takich przypadkach może Czytelnik zapoznać się z pracy Duby'ego [4] poświęconej symbolicznemu różniczkowaniu i znajdowaniu wzorów na pochodne w algebrach Lie'go.

Mimo różnych trudności, kierunek badań nad formalnym dokonywaniem przekształceń algebraicznych przez maszyny rokuje duże nadzieje na upowszechnienie i można sądzić, że ma przed sobą przyszłość.

Inny kierunek badań — to układanie tabel i wyliczanie elementów różnych algebr lub też obliczanie przez maszynę ich reprezentacji przez twory lepiej znane i łatwiejsze w opracowaniu, jak np. macierze. Prace takie prowadzone są w różnych ośrodkach i koncentrują się głównie na obliczaniu przy użyciu maszyn reprezentacji grup. Inspirowane są one przez potrzeby praktyczne, głównie przez potrzeby fizyki teoretycznej i wiążą się z katalogowaniem grup, tzn. wyliczeniem możliwie dużej liczby grup o danych własnościach. Wiele zagadnień dotyczących bądź to rozszerzeń grup, bądź znajdywania splotu grup ma już standardowe programy (zob. Leticzewskij [8], jak również Algorytm 136a [1]).

Wyznaczeniem na maszynach cyfrowych algebr innych niż grupy zajmowano się również w Polsce. Czyżo i Mostowski [3] podali algorytm znajdywania elementów wolnych struktur rozdzielnych o n generatorach, działający praktycznie na maszynie dla $n \leq 6$. Wyznaczenie wszystkich struktur rozdzielnych o ustalonej liczbie generatorów za pomocą maszyny mogłoby dać interesujące wyniki, o ile mi jednak wiadomo, badania takie nie były dotąd przeprowadzane.

Po tym szkicowym omówieniu różnych kierunków zastosowania maszyn w algebrze, omówimy szczegółowiej jeden kierunek, o którym dotąd nie wspominaliśmy, związany z badaniem grup przedstawionych przez generatory i relacje.

Najpierw kilka słów o przedstawieniu grup przez generatory i relacje. Przy takim przedstawieniu, grupę zadajemy przez podanie generatorów $x_1, \ldots, x_n$ i związków zachodzących między nimi zapisywanych w postaci

$$(1)\quad r_i(x_1, \ldots, x_n) = 1, \ldots, r_s(x_1, \ldots, x_n) = 1$$

gdzie r_i są słowami zbudowanymi z liter $x_1, \ldots, x_n$, opatrzonych wykładnikami potęg ± 1 za pomocą operacji mnożenia grupowego. Operuje się słowami nieskracalnymi, tzn. takimi, gdzie nie występuje kolejno żadna para $x_i^{+1} \cdot x_i^{-1}$ względnie $x_i^{-1} \cdot x_i^{+1}$ (równa 1 w każdej grupie).

Na przykład, zadając grupę przez dwa generatory x_1, x_2 z relacjami

$$x_1^2 = 1, \quad x_2^2 = 1, \quad x_1 x_2 = x_2 x_1 \quad (\text{czyli } x_1^{-1} x_2^{-1} x_1 x_2 = 1)$$

otrzymamy w wyniku grupę przemienną czteroelementową, w której kwadrat każdego elementu jest równy jedności grupy. Każde słowo, a ogólniej — wyrażenie zbudowane z x_1 i x_2 za pomocą działania grupowego można za pomocą relacji (jak wyżej) sprowadzić do jednego z czterech słów 1, x_1, x_2 lub $x_1 x_2$ przedstawiających różne elementy grupy.

Przedstawienie grup przez generatory i relacje ma już kilkudziesięcioletnią historię i wykorzystywane jest w topologii, fizyce molekularnej, krystalografii; budzi również z wielu względów zainteresowanie logików.

Czytelnika, który chciałby dokładniej zapoznać się z tym problemem odsyłam do przeglądowego referatu M. Hall [6], gdzie omówiona jest strona algebraiczna problemu, a także do książki Coxeter i Moser [2]. Ta ostatnia książka poświęcona jest związkom tych zagadnień z topologią i geometrią.

W związku z przedstawieniem grupy przez generatory i relacje powstaje cały szereg problemów, zarówno teoretycznych, jak i praktycznych. Oto one: Mając zadaną grupę G przez

$$(2)\quad \text{generatory: } x_1, \ldots, x_n \quad \text{oraz relacje } r_1(x_1, \ldots, x_n) = 1, \ldots, r_s(x_1 \ldots . x_n) = 1$$

chcemy mieć możność stwierdzenia, czy dwa słowa, tzn. wyrażenia zbudowane z $x_1, \ldots, x_n$ za pomocą działań grupowych mnożenia $x \cdot y$ i brania odwrotności x^{-1} są równe, czy nie, tzn. czy

$$(3)\quad f_1(x_1, \ldots, x_n) = f_2(x_1, \ldots, x_n)$$

dla słów f_1 i f_2 w grupie G czy nie.

Zagadnienie to nosi nazwę problemu słów.

Np. w grupie przedstawionej przez (1) słowa x_1 i $x_2x_1x_2$ przedstawiają ten sam element. Słowa $x_1x_2x_1$ i $x_1x_2x_1x_2$ przedstawiają różne elementy. Pierwsze równe jest w grupie G elementowi x_1, zaś drugie jedności grupy 1.

Drugie zagadnienie — to kwestia rozstrzygnięcia, czy grupa zadana przedstawieniem (2) jest skończona, czy też nie i ewentualnie w przypadku, gdy grupa ta jest skończona, wyznaczenie jej rzędu i ułożenia tabliczki „mnożenia" dla tej grupy. Zagadnienia te podciąga się pod wspólną nazwę problemu skończoności, rozumianego często węziej jako szukanie odpowiedzi, czy grupa przedstawiona przez (2) jest skończona, czy też nie.

Oba te bardzo ważne praktycznie zagadnienia nastręczały jednak wiele trudności, których istota została po części wyjaśniona dopiero przed kilkunastu laty. P.S. Nowikow udowodnił, że istnieją takie przedstawienia grupy, dla których nie istnieje algorytm pozwalający rozstrzygać dla dowolnych słów [3], czy przedstawiają one ten sam element grupy, czy nie.

Z prac jego i innych autorów (szczegółowe informacje znajdzie Czytelnik w M. Hall [6]) wynika, że istnieją przedstawienia grup, dla których nie istnieje algorytm pozwalający rozstrzygać, czy grupy zadane tymi przedstawieniami są skończone, czy nie. Mówi się krótko, że obydwa problemy są nierozstrzygalne. Zagadnienie to wiąże się w pewnym sensie z nierozstrzygalnością elementarnego rachunku kwantyfikatorów, o czym była mowa w części pierwszej naszego artykułu [1].

Jeżeli jednak grupa pokazana przedstawieniem (2) jest skończona, to istnieją algorytmy do stwierdzenia, czy dwa słowa (3) przedstawiają ten sam element, czy nie oraz algorytmy do wyznaczania rzędu grupy (tzn. liczby jej elementów) oraz układania tabelki. Metody te znane od około 30 lat polegają na tzw. numeracji warstw. Opiszemy je tutaj pokrótce.

Niech G będzie grupą o przedstawieniu (2), zaś H podgrupą generowaną przez elementy przedstawione słowami

$$g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n).$$

Rozpatrujemy warstwę, tzn. zbiory złożone z wszystkich iloczynów $h \cdot u$, gdzie $u \varepsilon G$ jest dane, a $h \varepsilon H$ przebiega całą podgrupę. Grupa G rozkłada się na sumę rozłącznych warstw. Każdy element grupy należy do jednej warstwy. Numerujemy najpierw warstwę elementu 1 (tzn. warstwę $H \cdot 1 = H$) np. liczbą 1, potem bierzemy np. generator x_1 i warstwę Hx_1 numerujemy liczbą 2. Biorąc element x_2 warstwę Hx_1x_2 numerujemy liczbą 3 i postępując tak dalej bierzemy wszystkie x_i oraz odwrotności x_i^{-1} numerując kolejne warstwy. Jeżeli np. rozpatrujemy mnożenie przez element x_2, a numer ostatniej warstwy $Hx_1x_2x_1$ jest 3, to warstwie $Hx_1x_2x_1x_2$ powinniśmy nadać numer 4. Jeżeli jednak wśród relacji przedstawiania (2) jest relacja $x_1x_2x_1x_2 = 1$, to warstwie tej nie możemy nadać numeru 4, bo $Hx_1x_2x_1x_2 = H \cdot 1 = H$ jest warstwą o numerze 1. Taka zmiana numeracji pociąga na ogół konieczność zmiany numeracji warstw poprzednio ponumerowanych (gdy są one równe).

Przestrzegając w opisanym postępowaniu pewne rygory (których nie opisujemy tutaj, aby nie wchodzić w szczegóły) ponumerujemy kolejno wszystkie warstwy grupy G względem podgrupy H.

Przeprowadzone postępowanie wskaże nam liczbę warstw grupy G względem grupy H, a więc znając liczbę elementów podgrupy H będziemy znali liczbę elementów grupy G równą liczbie elementów grupy H pomnożonej przez liczbę warstw. Postępowanie to pozwoli nam ponadto wyznaczyć numer warstwy $H.f(x_1, \ldots, x_n)$ dla dowolnego słowa $f(x_1, \ldots, x_n)$.

---

1) „Maszyny Matematyczne" nr 5/1969.

Gdy weźmiemy za H podgrupę składającą się tylko z 1 i przeprowadzimy numerację warstw, to odpowiedź na pytanie, kiedy dwa słowa (3) są równe otrzymamy już łatwo za pomocą prostego postępowania, bowiem rząd grupy G będzie równy liczbie warstw. Również tabelkę grupy będzie można łatwo ułożyć. Oczywiście, proces numeracji warstw nie skończy się, gdy grupa G będzie nieskończona, bo nieskończonego zbioru warstw nie możemy ponumerować w skończonej liczbie kroków. W przypadku, gdy G będzie grupą skończoną, zawsze uzyskamy interesujące nas odpowiedzi.

O metodzie numeracji warstw ukazało się wiele prac: Coxter-Moser [2], Todd [13], Leech [7], Mendelson [9]. Przeglądowe omówienie wyników związanych ze stosowaniem tej metody do zaprogramowania pracy maszyny znajdzie Czytelnik w pracy Leech [7]. Wydaje mi się, że najlepsze wyniki z opublikowanych dotąd, a nie przedstawionych w pracy Leecha, osiągnęli Felsch [5] i Trotter [14].

Pierwszy z nich ułożył program numeracji warstw dla maszyny Z 22, pozwalający znajdować rzędy, tzn. liczbę elementów dla grup o n generatorach dla rzędów nie przewyższających $5300/n$.

Szereg obliczeń zarówno do celów praktycznych, jak i teoretycznych wykazało użyteczność tego programu. Obliczenia wykonywano w czasach rzędu minut i godzin.

Trotter zmodyfikował znacznie proces numeracji warstw (przystosowany zasadniczo do obliczeń ręcznych) pod kątem jego uproszczenia z punktu widzenia maszyny. Ułożył on program pod nazwą COSET dla maszyny IBM 7090/94 do numeracji warstw i znajdywania przedstawienia generatorów grupy za pomocą permutacji warstw. Program wymagał, aby grupa miała $n \leqslant 9$ generatorów, a liczba warstw nie przekraczała $29\,000/_{n+2}$. Obliczenia wykonywane były niezwykle szybko, dla przykładu, dla grupy G generowanej przez x_1 i x_2 z relacjami $x_1^3 = 1$, $x_2^7 = 1$, $(x_1x_2)^2 = 1\,(x_1x_2^{-1})^3 = 1$ maszyna znalazła wszystkie 448 warstw względem podgrupy generowanej przez $x_1^2, x_1^{-1}x^2$ w czasie około 6 sekund. Program Felscha zużył na rozwiązanie tego samego problemu 4 godziny.

Niektórzy algebraicy, m. in. Marshal, Hall (zob. Todd [13]) wiązali z próbami podobnego typu duże nadzieje teoretyczne. Pewne sukcesy osiągnięto, i to nie tylko na polu obliczenia liczby elementów różnych grup, których rzędy nie były dotąd znane, ale również jeżeli chodzi o zagadnienia minimalizacji zbioru relacji określających.

Sposób postępowania jest w tym problemie niezwykle interesujący jako ilustracja powiązania pracy maszyny z pracą twórczą.

Jeżeli grupa G przedstawiona przez generatory $x_1, \ldots, x_n$ i relacje

$$(4) \quad r_1(x_1, \ldots, x_n) = 1 \ . \ . \ . \ , \ r_s(x_1, \ldots, x_n) = 1$$

jest skończona, to metodą numeracji warstw można wyznaczyć jej rząd. Można teraz badać, czy dana relacja $r\,x_1, \ldots, x_n = 1$ jest wnioskiem relacji (4), czy nie. Wymaga to obliczenia rzędu grupy o generatorach $(x_1, \ldots, x_n)$ i relacjach (4) uzupełnionych relacją $r(x_1, \ldots, x_n) = 1$. Szczegóły i informacje o uzyskanych w ten sposób wynikach znajdzie Czytelnik w pracy Leecha.

Uproszczono w ten sposób przedstawienie wielu grup. Próbowano również wyliczać rzędy grup mających K-generatorów $x_1, \ldots, x_K$, w których każdy element spełnia identycznie relację $x^4 = 1$, tzw. grup Burnside'a $B(K, 4)$ dla $K \geqslant 4$.

Wyniki tych prób nie są znane autorowi artykułu.

Inne badania tego typu dotyczyły struktury podgrup tzw. p-grup. Grupę G nazywa się p-grupą, jeżeli każdy element g należący do G ma rząd p^n, gdzie p jest

ustaloną liczbą pierwszą (zaś n może być zależne od elementu g). Prace takie wykonywa Neubüsser [11], [12] na maszynie Z22. Maszyna wyliczała kolejno wszystkie podgrupy p-grupy zadanej przez generatory i relacje oraz podawała diagram struktury podgrup, tzn. wskazywała wzajemne zawieranie się podgrup. Ze względu na pojemność pamięci i kłopoty techniczne związane z zapisem elementów p-grupy, obliczenia prowadzono tylko dla p-grup w przypadku $p = 2$. Elementy grupy dawało się wówczas opisywać słowami binarnymi. Maszyna wyliczała struktury dla szeregu 2-grup, mających kilkaset elementów. Struktury podgrup zawierały liczbę podgrup równą mniej więcej połowie liczby elementów (nie licząc podgrup cyklicznych) w czasie rzędu 5—8 godzin. Praca inspirowana była potrzebami teoretycznymi, związanymi z badaniami nad p-grupami.

Z innych wyników Autor niniejszego artykułu badał grupy nilpotentne (zob. A. W. Mostowski [10]), podał algorytmy w celu rozstrzygania problemu słów, problemu należenia do podgrupy i wyznaczenia rzędu grupy lub stwierdzenia, że grupa jest nieskończona, dla grup nilpotentnych o znanym stopniu nilpotentności. Algorytmy te opracowano pod kątem ewentualnej przydatności do zastosowania maszyn cyfrowych w ich realizacji. Praktycznie próby realizacji ich na maszynach zostały dopiero rozpoczęte.

Uwagi końcowe

Z najnowszym stanem prac nad zastosowaniem maszyn matematycznych w algebrze może się Czytelnik zapoznać z przeglądowego artykułu J. J. Cannona *„Computers in Group Theory (a jurvey), ogłoszonego* w *Communications of A.C.M.* 1969 pp. 3—12, już po napisaniu niniejszej pracy. Artykuł Cannona może być uważany za mniej więcej pełny przegląd publikacji i kierunków badań, jeżeli chodzi o zastosowanie maszyn m. in. w teorii grup. Lista cytowanych prac obejmuje około 80 pozycji uwzględniających różne problemy. Jej obszerność świadczy o szybkim rozwoju dyscypliny z pogranicza algebry i teorii maszyn matematycznych — przynajmniej, jeśli idzie o teorię grup. Liczba publikacji z zastosowań innych działów algebry jest przypuszczalnie o wiele mniejsza.

Z ogłoszonego przeglądu J. J. Cannona można wnioskować, że niektóre zagadnienia, jak np. zastosowanie maszyn do automatycznego dowodzenia twierdzeń, nie wspomniane w naszym artykule, ale wiążące się z zagadnieniami omawianymi w jego części pierwszej — znajdują się w stadium nieśmiałych prób. Trudno wnioskować, jaka będzie ich dalsza przyszłość; czy pozostaną one badaniami teoretycznymi, czy też znajdą praktyczne zastosowanie.

Inne natomiast zagadnienia — czy to pod wpływem potrzeb praktycznych (jak np. znajdowanie reprezentacji grup), czy też pod wpływem dużego zainteresowania badaczy teoretyków (jak np. numeracja warstw czy znajdywanie struktury podgrup) — zostały opracowane w sposób nadający się do praktycznego wykorzystania. Jeżeli chodzi o wyniki wiążące się z praktyką, czy to z fizyką czy krystalografią, to się już stosuje je praktycznie.

Inaczej wygląda sprawa z wynikami bardziej związanymi z teorią. Jak dalece algebraicy potrzebują tych wyników i czy skorzystają z nich w swojej pracy — za wcześnie jest mówić. Nie wiadomo obecnie, czy będą to tylko udane próby teoretyczne, czy też w przyszłości bliższej lub dalszej będziemy korzystać z maszyn matematycznych w sposób zbliżony do tego, w jaki konstruktorzy korzystają z nich przy wykonywaniu obliczeń konstrukcyjnych. Trzeba przyznać, że na razie na ogół brak jest zainteresowania tymi próbami nie tylko w starszych, bardziej tradycyjnych środowiskach algebraików, ale i wśród szerokiego grona specjalistów maszyn matematycznych. Być może — pierwsi zbyt sa zagłębieni w teorii, a drudzy — zbyt zafascynowani słowem użyteczność.

*

Na zakończenie Autor chciałby podziękować wszystkim tym, których uwagi, wskazówki, pomoc bibliograficzna, publikacje wewnętrzne lub rękopisy oraz rozmowy skłoniły go do napisania i umożliwiły opracowanie tej publikacji. Z żalem rezygnuję tutaj z wyliczenia Ich imiennie, nie chcąc znacznie przedłużać i tak niepomiernie już długiego artykułu.

BIBLIOGRAFIA

[1] Algorytm 136a, Razszirenija gruppy Z, Obszczyje woprosy programirowanija, algoritmi 101—150, 4(1967), tłum. z Communication of ACM

[2] H.M.S., W.O.J. Moser, Generators and Relations for Discrete groups, Ergebnisse der Mathematik

[3] E. Czyżo, A. W. Mostowski, Bull. Acad. Polon. Sci., ser. mat-fiz. Vol. XVI nr 7, 1968, p. 593—595

[4] J. J. Duby, Sophisticated algebra on a computer: Derivatives of Witt vectors. IFIP Working Conference Symbol Manipulation Languages, Pisa 1966

[5] H. Felsch, Programmirung der restklassen Abzählung eine Gruppe nach Untergruppen, Numerische Mathematik, 3(1961), 250—256

[6] M. Hall, Generators and Relations in Groups — The Burside Problems, Lectures on Modern Mathematics, Vol. II, Viley 1964, str. 42—92

[7] J. Leech, Coset enumeration on digital computers, Proc. Cambridge. Phil. Soc., 59(1963), 257—267

[8] Leticzewski, O wyznaczaniu na elektronowej maszynie cyfrowej obrazu splotu grup skończonych (w jęz. rosyjskim). Mat. Żurnal, 3/65

[9] N. S. Mendelsohn, An algorithmic solution for a word problem in group theory, Canadian Journal of Mathematics, 16(1964) 509—516

[10] A. W. Mostowski, Computational algorithmus for deciding some problems in special classes of groups. Fundamenta Math., 69(1966) 123—135

[11] J. Neubüsser, Bestimmungen der Untergruppenverbände endlicher p-gruppen auf einer programmgesteurten elektronischen Dualmaschine, Numerische Mathematik, 3(1961), 271—278

[12] J. Neubüsser, Untersuchungen des Untergruppenverbändes endlicher Gruppen, auf einer programmgesteuerten elektronischen Dualmaschine, Numerische Mathematik 2(1960), 280—292

[13] J. Todd, Survey of Numerical analisis, New York, 1962, 15, 534—538

[14] H. Trotter, An. algorithm for the Todd-Coxeter method of coset enumeration, Canadian Math. Bull, 7(1964), 357—368.

Prenumerujcie miesięcznik

„MASZYNY MATEMATYCZNE"

Władysław M. Turski
Centrum Obliczeniowe PAN
Warszawa

681.322.06

Obecny kryzys w dziedzinie oprogramowania maszyn cyfrowych[1)]

Autor stwierdza, że dotychczos nie udało się zbudować zadowalającej teorii oprogramowania podstawowego maszyn cyfrowych. Analizuje przykłady odmiennego traktowania przez matematykę i praktykę programowania problemów związanych z systemami liczącymi i wysuwa dezyderaty wobec teorii oprogramowania. W procesie budowy tej teorii trzeba będzie zrezygnować z pewnych metod matematyki, ale równocześnie należy wykorzystać jej dorobek.

„Kryzys, przesilenie, stan rzeczy poprzedzający zwrot ku dobremu lub ku złemu..." (Podręczny słownik języka polskiego, Warszawa, 1957, str. 116).

Oprogramowaniem podstawowym maszyny cyfrowej, a ściślej — systemu liczącego nazywać będziemy zespół środków programowych, umożliwiający wprowadzenie, wykonanie i przedstawienie wyników zadań przetwarzania informacji, w formie i trybie będących pochodnymi tego zespołu środków programowych w większym stopniu niż pochodnymi charakterystyk sprzętu cyfrowego, z którego złożony jest system liczący. Innymi słowy, oprogramowanie podstawowe — to wszystko to, co nie będąc sprzętem jest obecne w systemie liczącym, gdy wprowadzamy zadania użytkowe, gdy są one wykonywane, gdy są wyprowadzane ich wyniki, czego obecność jest przy tym niezbędna dla ustalonej formy i trybu pracy systemu, co wreszcie bardziej niż sprzęt określa jakościowe parametry systemu. Dla uproszczenia będziemy dalej posługiwali się skrótem OPS (Oprogramowanie Podstawowe Systemu)[2)].

Czy można powiedzieć, że panuje obecnie kryzys w dziedzinie OPS? Czy droga, jaką przebyliśmy od programowania tradycyjnych maszyn cyfrowych w kodzie wewnętrznym do programowania systemów wielodostępnych w przebogatej rozmaitości języków zewnętrznych, od podprogramów funkcji elementarnych do wyrafinowanych pakietów użytkowych, od elementarnych, szkieletowych monitorów do zarządzania programowego zasobami nader złożonych systemów krotnych, nie powinna nas skłaniać do zachwytu raczej niż do mówienia o kryzysie? Otóż — moim zdaniem — nie. Niemałe bowiem osiągnięcia w zakresie OPS noszą zbyt intuicyjny charakter, aby z niezbędną w sprawach naukowych rzetelnością powiedzieć można cokolwiek o ich ogólności. Pod presją tempa rozwoju techniki cyfrowej i wzrostu zapotrzebowania użytkowników na usługi OPS powstało zbyt wiele rozwiązań praktycznych, o których wiemy tylko tyle, że w zasadzie działają zadowalajaco, ale które okazują się nieprzydatne przy pozornie nieznacznych zmianach wymagań funkcjonalnych, jakie im stawiamy. W procesie budowy OPS nagromadziliśmy sporo doświadczenia, odkryliśmy wiele użytecznych tricków i metod skutecznego postępowania — ale, w zasadzie, doświadczeń tych nie potrafiliśmy — jak dotąd — skonsolidować w spójną i logiczną teorię budowy OPS, choćby nawet czysto fenomenologiczną, a odnośnie do metod cząstkowych, nie potrafimy jasno określić ich stosowalności.

Ludzi, którzy zawodowo mają budować OPS kształcimy prawie wyłącznie metodą „cechową", tj. przez terminowanie u „mistrzów". Wyniki pracy w tym zakresie opisujemy albo w postaci zbeletryzowanych opracowań, albo korzystamy każdy z własnej symboliki, dla innych niepojętej. Nie posiadamy, jak dotąd, nie tylko ogólnej podstawy, powszechnie przyjętej sumy wiadomości podstawowych, ale nawet niebardzo wiemy, jakie są pojęcia pierwotne i pierwotne obiekty OPS.

Wszystko to, a także szereg innych czynników, które pominiemy dla uproszczenia, skłania do poważnego zastanowienia się nad stanem wiedzy o OPS.

A konieczność życiowa, społeczna konieczność dalszej intensyfikacji prac w tej dziedzinie, konieczność zwiększenia wydajności pracy produkcyjnej — tworzenia dobrych OPS dla coraz to nowych zastosowań sprzętu cyfrowego — zmusza do rzetelnej analizy przyczyn tego stanu, aby znaleźć drogę poprawy. I, gdybyśmy powodowani dumą z niewątpliwych osiągnięć praktycznych, przymknęli oczy na nasze niedociągnięcia teoretyczne, to łacno moglibyśmy kontynuować pracę nadmiernym wysiłkiem, przy każdym nowym zadaniu powtarzając *ab ovo* całą pracę, nie znając nadal ani kryteriów jakości, ani zakresu stosowalności uzyskiwanych wyników.

To rozumowanie, którego esencję podałem powyżej, skłania mnie do określenia obecnej sytuacji w zakresie OPS — mianem kryzysu.

Maszyny cyfrowe — w naszym języku — nazywane są często maszynami matematycznymi. Nazwa ta zdaje się sugerować istnienie związku pomiędzy nimi i matematyką. Gdyby tak było istotnie, w matematyce — być może, należałoby szukać odpowiedzi na pytanie dotyczące podstaw OPS. Niestety, związek maszyn cyfrowych z matematyką nie jest ani tak mocny, ani tak głęboki, jak na to wskazuje, w gruncie rzeczy przypadkowe zestawienie nazw.

Maszyny cyfrowe, jakich używamy obecnie, są — w grubym uproszczeniu — wszystkie jednego typu; nazwiemy je dla skrótu myślowego, a także aby oddać należne człowiekowi, któremu tyle zawdzięczają — maszynami von Neumanna. Cechą charakterystyczną tych maszyn jest to, że realizują „elektroniczny analog" dobrze znanego z popularnych opracowań schematu — rachmistrz, arytmometr, tablice, formularz wyliczeń itd. W całym tym schemacie

---

1) Kierując niniejszy artykuł do druku, zdaję sobie sprawę z pozornej niecodzienności i kontrowersyjności niektórych tez w nim zawartych, uderzajacych zwłaszcza przy powierzchownym czytaniu. Spodziewajac się ostrej nawet dyskusji, liczę na to, że dyskutanci przełożą fachowość nad emocję.

2) Oczywiście OPS jest odpowiednikiem angielskiego "software". Byłoby może interesujące używać „ops" z rzeczownikową odmianą: ops, opsu, opsowi itd., zrezygnujemy z tego w niniejszym artykule tylko dlatego, aby nie powiększać nadmiernie ilości kontrowersji.

z matematyką jeden tylko element ma nieco wspólnego — formularz wyliczeń czyli algorytm, cała reszta jest *par excellence* empiryczna i, jeśli szukać teorii nią rządzącej, to znaleźć ją raczej można w rozważaniach prakseologicznych niż matematycznych.

Inaczej, oczywiście, miała by się cała sprawa gdyby maszyny cyfrowe, jakich używamy w praktyce, zbudowane były jako praktyczne realizacje, np. maszyny Turinga. Nie miałoby wówczas sensu kwestionowanie ich „matematyczności" z tego choćby względu, że ich działanie opisywałoby się zamkniętą teorią matematyczną, uwzględniającą dowolnie subtelny szczebel szczegółowości funkcjonalnej. Ponieważ powody, dla których taki stan rzeczy ma miejsce dla maszyn Turinga, a nie zachodzi dla maszyn von Neumanna są ogólnie znane, nie będziemy tego tematu rozwijać dalej.

Dla dalszych rozważań zapamiętamy tylko, że nagromadzenie w konstrukcji obecnych maszyn cyfrowych rozwiązań funkcjonalnych opartych o przesłanki empiryczne i pragmatyczne jest przyczyną tego, że nie są one z *natury rzeczy* realizacją systemu matematycznego. Tak więc, w najlepszym razie możemy mieć nadzieję na zbudowanie ścisłego *opisu matematycznego* maszyn cyfrowych tak, jak buduje się matematyczny opis innych systemów naturalnych. Pamiętać przy tym należy, że zazwyczaj opisy takie dotyczą pewnych idealizacji systemów naturalnych, chociażby ze względu na konieczność wprowadzenia założeń wstępnych, obejmujących ustaloną klasę rzeczywistych systemów i tym samym wyłączających inne systemy.

Od lat prowadzone są z niemałym wysiłkiem prace naukowe, którym przewodzi pragnienie zbudowania teorii maszyn cyfrowych. W tym kontekście warto zatrzymać się przez moment nad podstawowymi kierunkami pracy badawczej w zakresie maszyn cyfrowych.

Wyróżnimy następujące sześć kierunków:

1. Budowa sprzętu liczącego
2. Teoria automatów skończonych
3. Teoria wyliczalności
4. Budowa algorytmów użytkowych i jej teoretyczne podstawy
5. Teoria języków programowania i automatycznej translacji
6. Budowa oprogramowania podstawowego.

Powyższy wykaz został ułożony tak, że przedstawia drogę, która mogłaby połączyć dwa krańce praktyczne — budowę oprogramowania i budowę sprzętu liczącego.

Na dobrą sprawę, wszystkie cztery teorie „pośrednie" w tym wykazie powstały w mniej lub bardziej embrionalnej postaci, niezależnie od maszyn liczących, a nawet przed nimi. Ale nie ma chyba wątpliwości, że bardzo gwałtowny rozwój tych teorii w ostatnich dwóch dziesiątkach lat odbywał się pod hasłem „teoretycznej obsługi" maszyn cyfrowych. Niestety, mimo imponujących osiągnięć wyliczonych teorii, ich bezpośredni wkład do praktyki jest znikomy, z oczywistym wyjątkiem analizy numerycznej, traktowanej jako źródło algorytmów numerycznych. Ale nawet ta najbardziej praktycznie nastawiona z teorii nie objęła na szerszą skalę tak ważnych zagadnień, jak: generowanie algorytmów z opisu problemów, analiza cech wspólnych różnych algorytmów, wyodrębnienie pierwotnych obiektów i elementarnych operacji algorytmicznych; te właśnie zagadnienia są najbardziej interesującym aspektem teorii algorytmów z punktu widzenia budowy OPS.

W celu zilustrowania związku teorii „pośrednich" z rzeczywistymi maszynami cyfrowymi, niech mi będzie wolno posłużyć się cytatą dotyczącą jednej z nich:

„... Badania w zakresie automatów skończonych prowadzone były w nadziei zastosowania ich do maszyn cyfrowych. Myślę, że jest to nadzieja w zasadzie płonna, ponieważ skończoność automatów wykorzystywana jest, aby udowodnić, że automat w końcu powróci do [wybranego] stanu. Jednakże ktoś, kto czekać będzie, zanim IBM 7090 powtórzy stan wyłącznie dlatego, że jest automatem skończonym, musi przygotować się na bardzo długie czekanie" [1].

Teorie pośrednie stały się tymczasem pełnoprawnymi gałęziami matematyki i zaiste imponująca jest płodność drzewa, które powstało z tego szczepienia. Jednakże ceną akceptacji do kręgu wiedzy matematycznej, którą teorie te musiały zapłacić, było oderwanie się od ich podstaw empirycznych. Zamiast teorii opisujących z matematyczną klarownością i ścisłością realia systemów liczących powstały teorie, z matematyczną swobodą kreujące sobie przedmioty badań przez „niech będzie dane..."

Istotne problemy współczesnych systemów liczących, a bacząc na temat naszego artykułu — zwłaszcza problemy OPS — nie należą do tych problemów, którymi matematyka zajmuje się zwyczajowo, są jej obce i nie ma, jak dotąd, znaków, które by wskazywały na zmianę stosunku matematyków do tych problemów.

Kilka przykładów, jakie przytoczę, ilustruje tę tezę.

W praktyce OPS jednym z najbardziej podstawowych (i kłopotliwych) praw jest rozróżnienie zawartości „komórki" pamięci i jej adresu, przy czym zarówno jeden obiekt, jak i drugi, posiadają swe nazwy i często nazwa adresu bywa zawartością komórki o innej nazwie. Zatarcie ścisłego rozróżnienia „co jest czym" może doprowadzić nie tylko do błędnych logicznie programów, ale i do wysoce nieefektywnego wykorzystania systemu liczącego. Tymczasem dla matematyka „dokładne rozróżnienie pomiędzy obiektami matematycznymi i ich nazwami jest w praktyce wielce niewygodne [... i] niekonieczne, ponieważ nie powoduje żadnych nieporozumień, jeśli wszystkie zdania i wyrażenia są wystarczająco zrozumiałe" [2].

Dla matematyka zadanie pewnego zbioru X pociąga za sobą automatycznie zadanie zbioru 2^X wszystkich jego podzbiorów. Tymczasem, żaden z dwu praktykowanych sposobów zadawania zbiorów w OPS (i innych rodzajach programowania): wyliczenie i podprogram generujący elementy, nie pociąga za sobą automatycznie zadania jakiegokolwiek bądź podzbioru właściwego, gdyż każdy z nich musi być znowu zadany w taki sam sposób, tj. albo przez wyliczenie (skopiowanie wybranych elementów), albo przez nowy podprogram generujący (powstający w najlepszym razie przez modyfikację podgrogramu generującego zbiór wyjściowy). Co więcej, podczas gdy dla matematyka najmniej istotne głębokie spośród wszystkich własności systemu algebraicznego są zazwyczaj relacje pomiędzy jego elementami, w przeciwieństwie do relacji pomiędzy jego podzbiorami i funkcji na nim określonych" [3], dla człowieka parającego się rzemiosłem oprogramowania właśnie relacje pomiędzy elementami są nietylko z reguły bardzo istotne (relacje „następowania po", relacje numeryczne), ale także przysparzają wielu kłopotów praktycznych (porządkowanie efektywne).

Jeśli dodać do tego całkowity brak zainteresowania matematyki zbiorami skończonymi i nikły entuzjazm względem zbiorów nieskończonych, ale przeliczalnych, zobaczymy, jak różne są podstawowe zainteresowania matematyki i sztuki oprogramowania.

Nawet w dziedzinie algorytmów numerycznych, związek oprogramowania z dotychczasową matematyką nie jest dostatecznie głęboki. Co prawda, w ostatnich latach daje się zauważyć wzrost zainteresowania analizą dyskretną, pozwalający mieć nadzieję na wyeliminowanie „pomostu ciągłości", wprowadzanego przez klasyczną analizę pomiędzy dyskretne pomiary i dyskretne schematy rachunkowe, ale mimo to, jak dotąd, matematyka nie zbudowała podstaw analizy dyskretnej, zwłaszcza zaś tych podstaw najgłębszych: dyskretnych analogów pojęć przejścia granicznego, ciągłości i pochodnej. Ba, nawet pojęcie równości dwóch liczb w arytmetyce dyskretnej nie jest ustalone w sposób powszechnie zadowalający [4].

Przykłady odmiennego traktowania przez matematykę i przy oprogramowaniu podstawowych elementów teorii można by mnożyć dalej. Ograniczymy się jednak do zasygnalizowania tylko kilku z nich: matematyka traktuje funkcję (odwzorowanie) jako fakt zachodzący dla zadanych zbiorów — w oprogramowaniu mamy do czynienia z procesem, z *aktem dokonywania* odwzorowania elementów zadanych zbiorów. Matematyka rozwija teorię funkcji rekursywnych, traktując iteracje jako przypadek szczególny i mało interesujący — w praktyce programowania rekurencja, a zwłaszcza nieodłączny od niej proces podstawiania wyrażeń (w odróżnieniu od matematycznego faktu podstawienia) stanowią co najwyżej zło konieczne, rozwiązanie o znacznie mniejszych walorach praktycznych niż iteracja, nie wymagająca podstawień, a tylko zmianę wartości. Matematyka (wyjąwszy stosowaną), a przede wszystkim jej działy, co do których spodziewamy się, że z OPS powinny mieć najwięcej wspólnego: topologia, kombinatoryka, logika i metamatematyka, nie zajmują się analizą efektywności wykonywalnych działań, ograniczając się nader często do dowodu istnienia i wykonalności; dla praktyki, a także chyba i dla przyszłej teorii oprogramowania — sam fakt istnienia rozwiązania czy wykonalności (oprócz naturalnie dowodów konstrukcyjnych) nie często może być podstawą dalszego działania.

Z powiedzianego powyżej wynika — jak mi się wydaje — że nie jest ani przypadkiem, ani wynikiem nieudolności ludzi uprawiających teorie pośrednie, iż nie udało się dotąd zbudować zadowalającej teorii OPS. Taki stan rzeczy jest chyba konsekwencją pochopnie przyjętego założenia, że teorię taką można zbudować w oparciu o taką matematykę, jaka ukształtowała się dotychczas i systematycznego podporządkowywania empirycznej wartości budowanych teorii względom formalnej wyrażalności w języku tej matematyki.

Skoro więc dochodzimy do wniosku, że dotychczasowa matematyka nie może być bezpośrednią podstawą teorii OPS, to — czy trzeba „zwinąć choragiewkę", zrezygnować z wysiłku zbudowania takiej teorii? Chyba nie. Nie zdarzyło się jeszcze, o ile wiem, w historii naszej cywilizacji, aby teoria społecznie potrzebna nie została w końcu zbudowana.

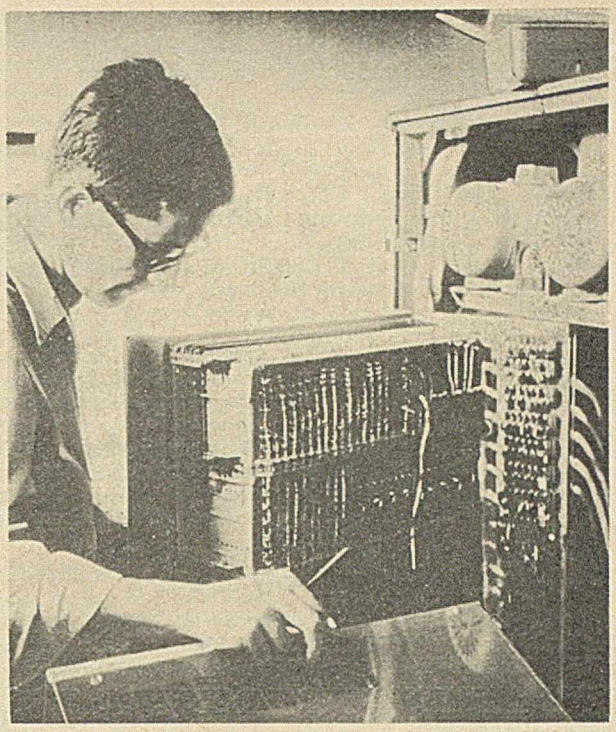
Wydaje się jednak, że aby taką teorię zbudować — a o jej potrzebie nie trzeba chyba nikogo przekonywać — należy się do tego zabrać bez „kompleksu niższości" względem istniejącej matematyki, nie starać się wyrażać odmiennych przecież problemów w jej dotychczasowym języku, ni próbować zmieścić się w ramach nakreślonych dla innych przecież zagadnień.

Należy przede wszystkim zająć się racjonalną analizą zgromadzonych doświadczeń, ustalić, jakie są elementarne pojęcia i obiekty OPS. Nie są nimi zapewne ani liczba, ani symbol, ani nawet stan. Być może są nimi proces i przejście od stanu do stanu, ze swymi atrybutami czasoprzestrzennymi.

Oczywiście nie miejsce tutaj, abym próbował definiować te pojęcia. Zresztą matematyka rozwijała się pomyślnie przez długie wieki, operując np. intuicyjnym pojęciem liczby i linii. *Nota bene* nawet tak intuicyjne pojęcie jak zbiór ma jednak inne zabarwienie w kontekście OPS niż w matematyce, na co wskazałem powyżej.

Należy ponadto ustalić, czego żądamy od teorii OPS, czemu ma ona służyć.

Oprócz złagodzenia tych bolączek, na jakie wskazywałem w pierwszych akapitach tego artykułu, nasuwają się jeszcze dwa główne dezyderaty względem teorii OPS: powinna dać ona możliwość dowodzenia, w matematycznym sensie twierdzeń o konkretnych OPS oraz pozwolić na jednolite traktowanie struktur



Jednostka centralna maszyny ZAM-41

informacyjnych i procesów ich przetwarzania, niezależne od środków sprzętowych użytych w konkretnym systemie liczącym.

Nie ulega przy tym, przynajmniej dla mnie, żadnej wątpliwości, że budowa takiej teorii OPS nie będzie działalnością pozamatematyczną, ani tym bardziej, antymatematyczną. Wręcz przeciwnie, byłoby zaprzeczeniem żywotności matematyki twierdzić, że nie zdoła ona wchłonąć teorii opartej na odmiennych, ale niesprzecznych — bo w pewnym sensie „ortogonalnych" założeniach, byłoby zaś całkowitą ignoracją twierdzić, że budując i rozwijając teorię OPS, można obejść się bez matematyki.

Tak więc przypuszczam, że zbudowanie teorii OPS, a raczej proces budowy tej teorii, aczkolwiek musi oderwać się od dotychczasowych podstaw i zrezygnować z pewnych metod matematycznych, powinien pozostać matematycznym w sensie ścisłości i dbałości o wewnętrzną spójność. Przypuszczam także, że wzbogaci on matematykę w równym stopniu, co ją wykorzysta.

Podziękowanie. Artykuł niniejszy oparty jest o referat pod podobnym tytułem, jaki wygłosiłem na seminarium przeglądowym w Centrum Obliczeniowym PAN w dniu 24 maja 1969 roku. Pragnę w tym miejscu podziękować wszystkim Kolegom, którzy wzięli udział w żywej dyskusji po moim referacie. Szczególnie dziękuję Profesorowi M. Warmusowi, którego głos w dyskusji przekonał mnie o celowości publikacji tez mego referatu dla sprawdzenia ich przyjęcia przez szersze grono zainteresowanych.

SPIS CYTOWANEJ LITERATURY

[1] J. Mc Carthy, Towards a mathematical Science of Computation, Proc. IFIP Congress 1962, p. 22
[2] H. Rasiowa, R. Sikorski, The Mathematics of Mathematics, PWN, 1963, p. 158
[3] P. R. Halmos, Lectures on Boolean Algebras, Van Nostrand Co., 1963, p. 17
[4] A. van Wijngaarden, Numerical Analysis as an Independent Science, BIT 6, (1966), p. 6.

Janusz Ilczuk
Zjednoczenie Energetyki
Warszawa

681.322.004.14

Cele i strategia zastosowań komputerów

Artykuł wyróżniony w KONKURSIE-69 ogłoszonym przez nasze czasopismo na temat „Najwłaściwsza tematyka przetwarzania informacji i organizacji sieci ośrodków obliczeniowych w kraju". Autor uzasadnia korzyści stosowania komputerów, głównie do usprawnienia procesów inwestycyjnych, kooperacji, wydajności pracy i eksportu. Wskazuje również na użyteczność automatyzacji, systemów ewidencyjnych i stosowania metod optymalizacyjnych. Postuluje, aby — obok rozwijania sieci ośrodków obliczeniowych branżowych i usługowych, powołać centrum obliczeniowe gospodarki narodowej.

Myli się ten, kto sądzi, że rozwój elektronicznej techniki obliczeniowej polega głównie na konstruowaniu coraz doskonalszych i szybszych komputerów, chociaż taki właśnie wniosek można wysnuć z dyskusji toczącej się od pewnego czasu na łamach prasy krajowej. Zbyt dużo mówi się i pisze o technicznych zaletach maszyn, a zwłaszcza o szybkościach wykonywania operacji — za mało o zastosowaniach i o tym, co komputery mogą i powinny przynieść gospodarce narodowej. Dlatego szczególnie ważne jest określenie celu wprowadzania elektronicznych maszyn cyfrowych w naszym kraju. Aby cel ten określić, trzeba sporządzić bilans codziennych problemów gospodarki i możliwości stwarzanych przez nowoczesne komputery [1]. Spróbujemy rozwiązać to zadanie na przykładzie kilku podstawowych problemów.

Problem nr 1 — Inwestycje

W referacie Biura Politycznego na II Plenum KC PZPR czytamy: „Najbardziej dotkliwym objawem zakłóceń w procesie inwestycyjnym jest wydłużanie cyklu budownictwa, zwłaszcza w budownictwie przemysłowym..."

Czy komputery mogą pomóc w rozwiązaniu tego problemu?

Kilka lat temu w jednym z polskich ośrodków obliczeniowych przeprowadzono próbę zbilansowania pewnej inwestycji przemysłowej za pomocą maszyny ODRA 1003. Próba wykazała, że plan budowy przewidywał zbyt mała liczbe monterów instalacji. Przy założonej ich liczbie budowa trwałaby nie 3 lata, jak planowano, ale około 15 lat. Nie chodzi tu o instalatorów lub innych specjalistów, bowiem ich liczbę można stosunkowo łatwo powiększyć już w trakcie budowy, gdy braki planu wyjdą na jaw. Równie dobrze plan może zawierać „niespodzianki" w zakresie niedostatecznej ilości maszyn budowlanych, materiałów, prefabrykatów lub środków transportu.

Chodzi natomiast o metodę, o to, że za pomocą elektronicznych maszyn cyfrowych można uniknąć zaskoczenia, a więc — improwizacji, interwencji, delegacji i całego olbrzymiego balastu trudności organizacyjnych, prowadzących do wydłużenia cyklu budowy, do przekroczenia planowanych kosztów, do naruszenia proporcji założonych w NPG. Każdą budowę, każdy plan można zbilansować na maszynie, można ujawnić błędy i luki organizacyjne jeszcze przed rozpoczęciem budowy. Pozwala to znaleźć odpowiednio wcześnie środki zaradcze, zapobiegające kosztownym niespodziankom na placu budowy.

Uniknięcie błędów w planach przy stosowaniu w pracach planistycznych „ręcznej techniki obliczeniowej" nie jest możliwe. Tu nie pomogą ani bodźce ekonomiczne, ani środki administracyjne. Człowiek nie jest w stanie panować nad potokiem informacji, które musi przetworzyć chcąc zbudować plan, harmonogram lub bilans. Musi go w tym wyręczyć maszyna. Komputery zastosowane na tym polu mogą przynieść wiele korzyści.

Cytowany referat — wśród wielu czynników wywierających ujemny wpływ na realizację inwestycji — wymienia „niedomogi w organizacji pracy i wykorzystaniu budowlanego sprzętu technicznego". I na tym odcinku komputery mogą pomóc. Poczynając od szybko, dokładnie i bezbłędnie opracowywanych planów operacyjnych, a kończąc na wyznaczaniu drogi krytycznej, codziennie od nowa przeliczanej tak, aby kierownik budowy, szef bazy transportowej, zaopatrzeniowcy i magazynierzy, majstrowie i brygadziści wiedzieli w porę i dokładnie, co mają w danym dniu do zrobienia i na jaki odcinek pracy muszą zwrócić szczególną uwagę.

Szczególnie obiecujące jest powiązanie metod sieciowych z planowaniem kroczącym, które dotychczas stosowano w przemyśle, a które należałoby upowszechnić w budownictwie. Metoda planowania kroczącego polega na comiesięcznym lub częstszym aktualizowaniu planów operatywnych stosownie do wyników osiąganych bieżąco. Komputery pozwalają na opracowywanie stale aktualnych planów operatywnych, sporządzanych z największą dokładnością dla każdego wyrobu, detalu — dla każdego stanowiska pracy, dla każdego, kto uczestniczy w realizacji planu. Pozwalają sporządzać szczegółowe plany zaopatrzenia materiałowego, wyprzedzające znacznie okres wykonawstwa.

Zastosowanie opisanych tu środków może w istotny sposób poprawić organizację robót budowlanych i zapewnić terminową realizację zadań.

Drugi problem nr 1 — Kooperacja

Trudno nazwać kooperację problemem nr 2. Jest on równie ważny jak inwestycje. Stąd określnie: drugi problem nr 1.

Narzekania na niesolidność kooperantów są powszechnie znane. Problem kooperacji wybucha raz po raz, jak niespokojny wulkan, przysłaniając wszystkie inne mankamenty funkcjonowania gospodarki. Sposo-

[1] Używamy tu na przemian określenia „komputer" i „elektroniczna maszyna cyfrowa", używając obydwa określenia za równoznaczne. Korzystanie z obydwóch tych określeń jest celowe ze względów stylistycznych.

by, jakimi usiłuje się zapobiec niesolidności kooperantów nie odpowiadają potrzebom, ani pokładanym w nich nadziejom. M. in. stosuje się publiczne napiętnowanie w prasie, radiu i telewizji zupełnie tak, jakby zahamowania dostaw wynikały z opieszałości, braku odpowiedzialności lub ze złej woli kierownictwa albo załogi kooperanta. Napiętnowanie działa według zasady: zawstydźmy go to się poprawi. Recepta ta jest równie skuteczna, jak egzorcyzmy lub zaklęcia znachora.

Tymczasem jedynym skutecznym i adekwatnym do rangi problemu sposobem jest ścisłe kontrolowanie przebiegu produkcji w zakładzie dostawcy, prowadzone za pomocą elektronicznych maszyn cyfrowych. Dokładny i zbilansowany wewnętrznie plan produkcji i kontrola wykonania zadań oparta o metody sieciowe takie, jak PERT, CPM i inne stosowane na co dzień, systematycznie i powszechnie mogą być skutecznym środkiem pozwalającym „niesolidnym" kooperantom na terminowe wywiązywanie się z przyjętych zobowiązań. Żeby jednak tę doskonałość osiągnąć, trzeba przeliczać drogę krytyczną codziennie, trzeba badać i usuwać przyczyny utrudniające wprowadzenie nowoczesnych metod zarządzania — do praktyki.

Omawiając problem planowania operatywnego, warto zwrócić uwagę na jedno z ciekawych osiągnięć ZETO: metodę „rozwijania" wyrobu złożonego.

Dla jednej z fabryk opracowano metodę „rozwijania silnika". Polega ona w ogólnym zarysie na tym, że za pomocą komputera oblicza się i sporządza specyfikację zespołów, podzespołów i kompletu detali dla każdego typu silnika. Jeśli więc plan produkcji zakłada wykonanie pewnej liczby silników określonego typu, to maszyna cyfrowa może w tempie nieosiągalnym dla najlepszego planisty obliczyć i wyspecyfikować pełną listę elementów składowych wyrobu. Pozwala to na dokładne i — co najważniejsze — szybkie planowanie produkcji detali, zbilansowanie zdolności produkcyjnej maszyn, określenie potrzeb materiałowych i robocizny. Pozwala wreszcie dostatecznie wcześnie ulokować zamówienia u kooperantów tak, aby mogli we właściwym czasie opracować swoje plany produkcyjne, zaopatrzenia itp.

Szybkość i dokładność oraz wielostronność opracowań uzyskiwanych z komputera jest atutem, dzięki któremu może on stać się najskuteczniejszym lekiem na dolegliwości kooperacji.

Trzeci problem nr 1 — Wydajność pracy

Od szeregu lat roczne plany zatrudnienia są przekraczane. Od szeregu lat przyrost produkcji osiągamy w zbyt dużym stopniu kosztem wzrostu zatrudnienia, zamiast w rezultacie wzrostu wydajności pracy. Powoduje to szereg ujemnych skutków, prowadząc m.in. do trudności w zrównoważeniu popytu i podaży na artykuły pierwszej potrzeby. Podejmowane dotychczas różnorodne akcje, włącznie z wprowadzeniem norm technicznie uzasadnionych przyniosły pewien postęp, ale nie rozwiązały problemu na dłuższą metę. Dzieje się tak dlatego, że stosowana dziś technika ewidencji i planowania robocizny nie odpowiada wymaganiom nowoczesnej gospodarki. I tu właśnie olbrzymie pole do popisu mają komputery. I to nie poprzez obniżenie zatrudnienia pracowników biurowych, ale przede wszystkim i w przeważającej mierze — poprzez obniżenie nakładów pracy fizycznej.

Przykładem niech służy doświadczenie ZETO—Wrocław, a zwłaszcza system SYKOPP, który pozwala obliczać:

- „pracochłonność wyrobów w rozbiciu na kategorie robót i wydziały,
- pracochłonność oraz koszt robocizny bezpośredniej wyrobów, rozdzielone na wydziały, kategorie robót i grupy zaszeregowania"[2]).

Nie trzeba dodawać, jak duże znaczenie dla kontroli robocizny i zarobków mają podane tu wyliczenia. Nic też dziwnego, że w fabryce, w której zastosowano system SYKOPP,

„w wyniku wyliczeń aktualnych normatywów czasowych oraz wykorzystaniu tych danych do planowania produkcji, a co za tym idzie — ustalania wymiernych zadań dla wydziałów i stanowisk — uzyskano dodatkowy przyrost produkcji rzędu 10 mln zł rocznie"[3]). Zastosowanie komputerów do szczegółowego planowania robocizny, kontrola pracy i zarobków za pomocą zautomatyzowanej ewidencji robocizny dają realną możliwość bieżącego kontrolowania wydajności pracy, pozwalają unikać improwizacji w organizowaniu pracy, zapewniają szybkie wykrywanie przyczyn niedostatecznej wydajności, przestojów i innych źródeł strat czasu.

Elektroniczne maszyny cyfrowe są więc jedynym skutecznym narzędziem kontroli wykonywania planów wzrostu wydajności pracy, a przy tym stanowią niezwykle skuteczny środek wykrywania rezerw.

Czwarty problem nr 1 — Eksport

Eksport jest tym czynnikiem, który w najbardziej syntetyczny sposób pozwala mierzyć nowoczesność i jakość produkcji, stopień opanowania postępu technicznego i rozpowszechnienia postępu w przedsiębiorstwach. Kontakty zawierane przez nasze centrale handlu zagranicznego nie powstają wyłącznie jako rezultat działalności handlowców. Korzenie powodzenia w walce konkurencyjnej na rynkach zagranicznych tkwią w biurach konstrukcyjnych i projektowych. Jeśli współczesny konstruktor lub projektant chce, aby wyroby, nad którymi pracuje, osiągnęły poziom światowy, nie może posługiwać się tylko suwakiem logarytmicznym, rysownicą i własną inwencją. Tak uzbrojony będzie przypominał człowieka wyposażonego w łuk i strzały, któremu przyszło toczyć walkę z nieprzyjacielem uzbrojonym w nowoczesną szybkostrzelną broń palną.

Współczesny konstruktor bowiem powinien być wyposażony w pióro świetlne i wideograf sprzężony z komputerem, wykonującym obliczenia bieżąco. Może on szybciej i lepiej tworzyć projekty, rozpatrywać i przeliczać liczne warianty konstrukcji, stosować matematycznie uzasadnione, optymalne rozwiązania. Tak opracowane elementy konstrukcji rysuje automatyczny kreślarz, sterowany przez komputer.

Opracowane w opisany sposób konstrukcje mogą zapewniać lekkość, wytrzymałość i precyzję. Oznacza to nie tylko sukcesy handlowe, ale także oszczędność materiałów i polepszenie jakości wyrobów.

Te same urządzenia zastosowane w biurach projektowych mogą niezwykle korzystnie oddziaływać na koszty inwestycji i zlikwidować opóźnienia w wykonywaniu dokumentacji projektowej.

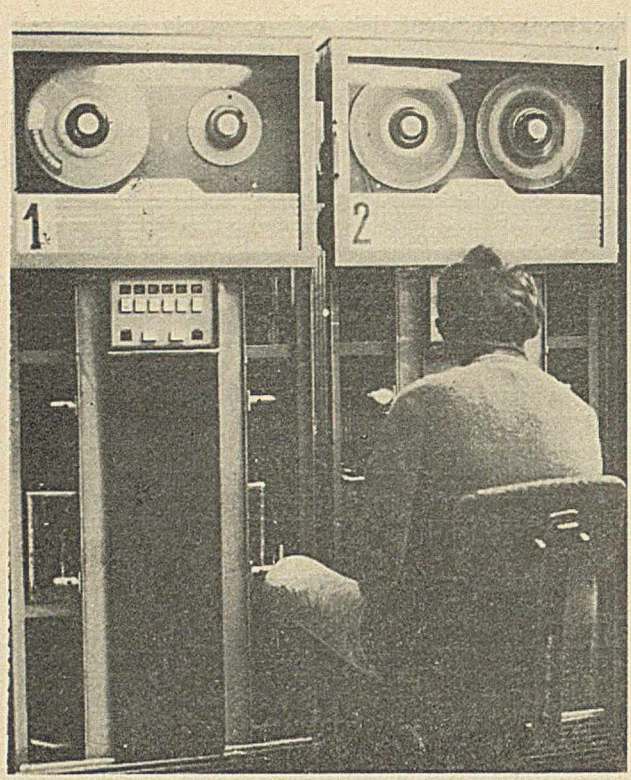
Konstruktor, projektant, pracownik instytutu i działacz gospodarczy musza na bieżąco zaznajamiać sie z literatura fachową, powinni z ośrodków informacji otrzymywać szybko zestawienia interesujacej ich literatury. Ręcznie prowadzone katalogi, podobnie jak ręcznie sporządzane zestawienia bibliograficzne nie sa w stanie zapewnić ani szybkości, ani kompletności informacji. Dlatego też i na tym polu zachodzi pilna konieczność zastosowania komputerów, tym bardziej, że we wszystkich wysoko uprzemysłowionych krajach już się je stosuje do opisanych tu celów.

To samo dotyczy ewidencji i udostępniania opisów patentowych.

Trudno w krótkiej wypowiedzi opisać wszystkie problemy gospodarki, które niemal w 100% można rozwiązać przez umiejętne stosowanie komputerów. Lista tych problemów nie kończy się bynajmniej na wymienionych dotychczas węzłowych zagadnieniach gospodarki.

2) A. Ramułt, L. Wolański — „System kontroli i planowania produkcji SYKOPP-1", „Maszyny Matematyczne" nr 4/69.

3) Ditto.



Pamięci taśmowe polskiego komputera ZAM-41

Można wskazać szereg innych: oto np. systemy ewidencyjne zastępujące księgowość i nie tylko księgowość, mogą skutecznie chronić mienie społeczne, uniemożliwiają kradzieże i nadużycia. Zastosowanie komputerów jest tu znacznie skuteczniejsze niż najsurowsze rygory dla księgowych i kierowników odpowiedzialnych „za brak nadzoru".

Automatyzacja ewidencji materiałów jest niezwykle cennym narzędziem likwidacji zespasów ponadnormatywnych i zbędnych, a także nie mniej skutecznym sposobem zmniejszenia zapasów bieżących. Przeprowadzona w dostatecznie dużym kręgu zakładów może spowodować likwidację braków materiałowych, skrócić przestoje w fabrykach i na budowach, a z kolei przyczynić się do terminowego wykonania zadań przez wykonawców (inwestycje) i przez kooperantów (przemysł).

Mówiąc o ewidencji i jej automatyzacji nie sposób pominąć tak istotnego momentu, jak to, że właśnie wprowadzenie zautomatyzowanych systemów ewidencji źródłowej we wszystkich dziedzinach działalności gospodarczej powoduje powstanie banku danych, stanowiącego fundament systemów informacyjno-decyzyjnych i dostarczającego niezbędnego w zarządzaniu opartym o metody matematyczne zestawu podstawowych informacji, podlegających przetworzeniu. Trzeba pamiętać, że bez danych jakiekolwiek przetwarzanie nie ma sensu. Tymczasem zarejestrowanie danych źródłowych na maszynowych nośnikach informacji zamienia to, co nazywa się czasem cmentarzyskiem liczb w kwitnący życiem, zawsze aktualny i zawsze dostępny zbiór podstawowych informacji.

Zastosowanie metod optymalizacyjnych, opartych na aktualnych zbiorach informacji, może mieć również wielorakie, i jak to udowodniły polskie doświadczenia — bardzo korzystne wyniki.

Np. optymalizacja przewozów może znacznie podnieść zdolność przewozową środków transportu kolejowego, samochodowego, rzecznego i morskiego.

W rolnictwie optymalizacja rozdziału nawozów sztucznych może zapewnić znaczny wzrost plonów podobnie, jak i optymalizacja struktury zasiewów.

Wreszcie, pomijając szereg bardzo istotnych zastosowań komputerów — szereg równie długi, jak długa jest lista dziedzin naszej gospodarki, wymienić trzeba problem stworzenia w przyszłości ogólnokrajowego systemu bieżącej ewidencji, statystyki i planowania opartego na elektronicznej technice obliczeniowej. Przecież gdybyśmy nawet bez jakichkowiek ulepszeń zastosowali dziś komputery do ewidencji i przetwarzania danych statystycznych, które codziennie płyną szerokim strumieniem z zakładów pracy do zjednoczeń, stąd — do resortów i dalej — do organów decydujących, moglibyśmy znakomicie usprawnić proces codziennego zarządzania gospodarką. Moglibyśmy bezbłędnie bieżąco eliminować informacje nie wymagające decyzji oraz również bieżąco i bezbłędnie specyfikować i charakteryzować liczbowo wszystkie odcinki zagrożone, wymagające interwencji ze strony różnych szczebli aparatu zarządzania.

W ten sposób moglibyśmy codziennie wiedzieć, w jakim stopniu wykonywane są w skali krajowej zadania NPG, gdzie powstają zakłócenia, a gdzie rezerwy i nadwyżki, które bieżąco można zagospodarować z wielką korzyścią dla gospodarki narodowej. Moglibyśmy wówczas nasze narodowe plany gospodarcze, sporządzane na rok następny np. w listopadzie, opierać na wykonaniu za 10 miesięcy, a nie za 5 miesięcy, a więc planować o wiele dokładniej.

Powie ktoś, że to fantazja. W epoce chrzęstu drewnianych liczydeł — przedstawione tu projekty mogą istotnie wyglądać na fantazję, ale w zbliżającej się erze zastosowań komputerów rzeczywistość pokaże, że to, co tu zaproponowano, było jedynie nieśmiałym, nie wyczerpującym ułamka rzeczywistych możliwości spojrzeniem w przyszłość.

Zważmy, że to człowiek decyduje, do czego i w jakim celu zastosuje się te najdoskonalsze narzędzia zarządzania. Zważmy, że socjalizm stwarza szczególnie dogodną sytuację do nie skrępowanych, powszechnych zastosowań komputerów w interesie gospodarki narodowej. Zważmy wreszcie, że dziś nikt nie jest w stanie przewidzieć tych wszystkich możliwości, jakie niosą ze sobą komputery — podobnie, jak nikt nie był w stanie przewidzieć olbrzymiego wachlarza zastosowań elektryczności wówczas, gdy instalowano pierwsze prądnice do oświetlania ulic.

Strategia rozwoju zastosowań komputerów powinna wyrastać z celów zastosowań. Omówione tutaj kierunki zastosowań sugerują ich powszechność. Trzeba więc zapewnić szeroki dostęp do tych maszyn wszystkim zainteresowanym. A zainteresowane są przede wszystkim jednostki gospodarki oraz wyższe uczelnie, których niezwykle pilnym zadaniem staje się kształcenie specjalistów, przygotowanych i przyzwyczajonych do codziennego posługiwania się — w ramach swojej specjalności — komputerami jako nieodzownym narzędziem pracy. Aby spełnić tak sformułowany postulat, trzeba rozwijać sieć ZETO. Maszyny ZETO powinny w pełni, a nie tylko dla wzbudzenia entuzjazmu zaspokajać potrzeby tych jednostek gospodarczych i uczelni, które nie mogą efektywnie w pełni obciążyć własnych maszyn. Oprócz ZETO trzeba rozwijać sieć ośrodków branżowych, obsługujących ważniejsze dla gospodarki branże, dostatecznie przygotowane do obciążenia maszyn.

Trzeba wreszcie powołać szczególnie dobrze wyposażone centrum obliczeniowe gospodarki narodowej, które powiąże ośrodki usługowe i branżowe, zapewni koordynację rozwoju zastosowań komputerów w skali krajowej drogą współpracy technologicznej, a nie administracyjno-biurokratyczną i które obsłuży centralne organa zarządzające gospodarką narodową.

Reasumując, należy podkreślić, że koncepcje rozwoju sieci ośrodków obliczeniowych w kraju muszą wyrastać z kierunków zastosowań, muszą opierać się na zasadzie maksymalnej użyteczności do potrzeb rozwoju gospodarki narodowej w szerokim znaczeniu tego słowa. Dopiero na podstawie tak określonych zadań można dobierać maszyny, można rozumieć związek pomiędzy pogonią za szybkością operacji a codziennym życiem.

Stefan Andrejszyn
Centralny Ośrodek
Badawczy Normalizacji
Warszawa

681.322.004.14:389.64

Mgr Stefan Andrejszyn ukończył Wydział Matematyczno-Przyrodniczy na Uniwersytecie Warszawskim. Pracował w Państwowym Zakładzie Emerytalnym, w Centralnym Zarządzie ZOR, a od 1955 r. w Państwowej Komisji Planowania Gospodarczego, a po przemianowaniu — w Komisji Planowania przy Radzie Ministrów. W 1963 r. zostaje adiunktem naukowo-badawczym i kierownikiem Pracowni Programowania Matematycznego. W 1965 r. został mianowany samodzielnym pracownikiem naukowo-badawczym w Instytucie Planowania oraz Centrum Obliczeniowym KP przy RM, którego był jednym ze współorganizatorów. Od 1968 r. pracuje w Centralnym Ośrodku Badawczym Normalizacji na stanowisku samodzielnego pracownika naukowo-badawczego i jest kierownikiem Pracowni Metod Elektronicznej Techniki Obliczeniowej. Jest autorem wielu prac i publikacji.

Technika obliczeniowa w zastosowaniu do zagadnień normalizacji

Zastosowanie elektronicznej techniki obliczeniowej do porównywania norm przedmiotowych PN i BN z odpowiednimi normami innych krajów

Omówiono problem zastosowania komputerów do porównywania norm przedmiotowych polskich z odpowiednimi normami innych krajów. Opracowaną metodę wypróbowano na przykładzie norm gatunków stali węglowych konstrukcyjnych pięciu krajów; programy na elektronicznych maszynach cyfrowych MIŃSK-22 opracowano w Centralnym Ośrodku Badawczym Normalizacji i uruchomiono w ZETO-Katowice.

1. OGÓLNA CHARAKTERYSTYKA PROBLEMU

Jak powszechnie wiadomo, elektroniczna technika obliczeniowa zastosowana początkowo do bezpośrednich żmudnych obliczeń numerycznych znajduje coraz więcej zastosowań, wkraczając nawet do takich dziedzin, które były uważane za domenę jedynie działalności umysłowej człowieka. Elektroniczna maszyna cyfrowa może bowiem nie tylko wykonywać operacje arytmetyczne, ale również i operacje logiczne takie, jak sumowanie i mnożenie logiczne, porównywanie itd.

Normalizacja, której jednym z głównych zadań jest porządkowanie pewnych zbiorów obiektów i przepisów, stanowi tę dziedzinę, gdzie elektroniczna technika obliczeniowa będzie prawdopodobnie dość szeroko wykorzystywana — właśnie ze względu na zdolność maszyn cyfrowych do wykonywania operacji logicznych.

W Centralnym Ośrodku Badawczym Normalizacji (COBN) podjęto próbę zastosowania elektronicznej maszyny cyfrowej do porównywania polskich norm przedmiotowych z odpowiednimi normami innych krajów. Celem tematu było wypracowanie metody porównywania norm różnych krajów. Weryfikację metody postanowiono przeprowadzić na przykładzie porównywania polskich i zagranicznych norm gatunków stali węglowych konstrukcyjnych. Jest to temat eksperymentalny, pomyślany jako problem wycinkowy w przyszłym kompleksowym systemie elektronicznego przetwarzania informacji normalizacyjnych.

Problem porównywania norm jest ważny dla wielu dziedzin przemysłu i handlu. Potrzeba szybkiego porównywania norm występuje w pracach central handlu zagranicznego w realizacji zakupów dla odbiorców krajowych, którzy żądają produktów spełniających określone parametry techniczne, jak i realizacji zamówień odbiorców zagranicznych. Jest ono ważne dla biur projektów i innych instytucji przy zakupie licencji, przy adaptacji licencji lub korzystaniu z zagranicznych materiałów informacyjnych we własnych pracach ośrodków projektowych i badawczych. Wreszcie porównywanie norm daje elementy decyzyjne dla normalizatorów, którzy uzyskują dzięki temu obszerną informację wyjściową dla swojej pracy.

Praktycznego znaczenia problemu dowodzi chociażby fakt, że m.in. przemysł obrabiarkowy, który ma rozpocząć produkcję obrabiarek zespołowych w ramach kooperacji z Niemiecką Republiką Demokratyczną, zwrócił się z tym zagadnieniem do COBN. Jest to więc zagadnienie ważne przy rozwijaniu kooperacji w ramach RWPG.

Sposób porównywania norm zależy od tego, jaką formę zapisu przyjęto dla danej normy. Jak wiadomo, w praktyce wykorzystywane są trzy formy zapisu:

- wzór matematyczny
- opis słowny
- opis za pomocą parametrów.

Jeżeli chodzi o formę zapisu w postaci wzoru matematycznego daje się ona zastosować tylko w niektórych przypadkach. Jest to szczególny przypadek normy opisanej za pomocą parametrów, gdzie za pośrednictwem wzoru matematycznego możemy sprowadzić szereg parametrów do jednego. W tym przypadku nie ma problemów z porównywaniem norm przy wykorzystaniu elektronicznej maszyny matematycznej. Wystarczy zaprogramować wzór matematyczny i wykorzystując program porównywania liczb — uzyskać z łatwością porównanie norm.

Porównywanie norm w przypadku opisu słownego jest w chwili obecnej zagadnieniem trudnym przy istniejących maszynach cyfrowych. Aby można było porównywać teksty, muszą być one podane w językach deskryptorowych. Język taki ma wszelkie cechy kodu sformalizowanego. Jego syntaksa i leksyka różni się tylko postacią znaków od np. kodu wzorów matematycznych.

Wprawdzie skonstruowanie języka deskryptorowego do potrzeb normalizacji jest możliwe, będzie to jednak praca bardzo żmudna. Aby bowiem skonstruować język deskryptorowy opisu norm danej dziedziny, należy:

- wydzielić zakres przedmiotowy języka deskryptorowego, czyli skończoną listę przedmiotów lub cech, które będziemy chcieli opisywać za pomocą danego języka,
- skonstruować skończoną listę znaków dla tych cech i przedmiotów, przeprowadzić szczegółową analizę ich poprawności i efektywności z punktu widzenia wykorzystywania tych znaków,
- ustalić zakres pól semantycznych deskryptorów i predykatów języka (pola te powinny być „ostre", a przynajmniej „dostatecznie ostre"),
- skonstruować słowniki tezaurusowe — w zależności od potrzeb terminologiczne lub semantyczne,
- opracować procedury przetwarzania w sformalizowanym metajęzyku.

Dopiero wówczas możemy uznać, że dysponujemy językiem, który umożliwia zastosowanie elektronicznej techniki obliczeniowej do porównywania opisów zawartych w normie.

W praktyce do opisu normy wykorzystywany jest polski język etniczny. Język ten nie posiada wszystkich niezbędnych cech języka deskryptorowego. Stąd też w chwili obecnej porównywanie norm opisywanych słownie trzeba uznać za niewykonalne za pomocą komputera. Należy dodać, że do porównywania tekstów potrzebne są maszyny cyfrowe o bardzo dużych pamięciach i bardzo dużej szybkości. Powinny to być pamięci o dostępie swobodnym. W chwili obecnej dysponujemy w Polsce jedną maszyną posiadającą pamięć dyskową (IBM 1440), jednak o wiele za wolną, aby można było prowadzić na niej badania nad efektywnym porównywaniem tekstów. Dlatego w obecnym stadium rozwoju języków deskryptorowych i elektronicznej techniki obliczeniowej, problem automatycznego porównywania norm opisywanych słownie trzeba uznać za nie nadający się do praktycznego rozwiązania, chociaż w przyszłości na pewno powstaną warunki do realizacji takich prac.

2. SFORMUŁOWANIE TEMATU

Z wymienionych powodów skoncentrowano się na problemie automatycznego porównywania norm opisywanych za pomocą parametrów.

Mówiąc językiem algebry wektorów, każdą normę opisywaną przez skończoną liczbę parametrów możemy przedstawić w postaci wektora n-wymiarowego, w którym każda ze składowych będzie interpretowana jako określony parametr.

Porównanie norm sprowadzać się będzie do porównywania wektorów.

Przypuśćmy, że mamy dwa produkty X i Y opisane za pomocą wektorów parametrów

$$X = (x_1, ..., x_n)$$

$$Y = (y_1, ..., y_n)$$

Produkty uznamy za identyczne, jeżeli dla wszystkich $i = 1, ..., n$ jest spełniona równość

$$x_i = y_i$$

W praktyce normalizacji zagadnienie jest bardziej skomplikowane, ponieważ dopuszczalne są pewne odchylenia parametrów od żądanej wielkości, czyli parametr może się zmieniać w pewnym przedziale. Powinny być zatem spełnione warunki

$$x_i^- \leqslant x_i \leqslant x_i^+$$

$$y_i^- \leqslant y_i \leqslant y_i^+$$

gdzie

x_i^-, y_i^- — najmniejsza dopuszczalna wartość parametrów odpowiednio x_i i y_i.

x_i^+, y_i^+ — największa dopuszczalna wartość parametrów odpowiednio x_i i y_i,

dla wszystkich $i = 1, ..., n$. Jak więc widać dla porównania dwóch norm z dwóch różnych krajów będziemy mieli $2n$ warunków ograniczających.

W ten sposób zapis formuły porównywalności dwóch norm będzie następujący:

I przypadek

- produkt X jest zamiennikiem produktu Y wtedy i tylko wtedy, gdy każdy z parametrów charakteryzujących produkt X zmienia się w przedziale zmienności odpowiadającego mu parametru produktu Y

II przypadek

- produkt X jest zamiennikiem produktu Y, jeżeli każdy z parametrów charakteryzujących produkt X przyjmuje przynajmniej jedną taką wartość, która mieści się w przedziale zmienności odpowiadającego mu parametru produktu Y.

Jak z powyższych warunków widać, porównywalność norm nie dostarcza informacji o wzajemnym zastępowaniu się produktów.

Z faktu, że wyrób X jest zamiennikiem wyrobu Y, nie wynika, że wyrób Y jest we wszystkich przypadkach zamiennikiem wyrobu X. Niemniej jednak mówi, że przynajmniej w jednym przypadku, dla którego sporządzaliśmy porównanie norm, produkty te są wzajemnie zastępowalne.

Ogólna forma zapisu porównywania norm jako porównywania wektorów jest przydatna dla wszystkich kryteriów porównywania, jeżeli tylko informacja zawarta w parametrach stanowiących normę wystarcza do tego, aby dokonać porównania z punktu widzenia wybranego kryterium. I tak, możemy dowolnie zmniejszać bądź zwiększać liczbę parametrów, zmieniać przedziały ich zmienności. Formuły (I) i (II) będą prawdziwe dla dowolnej liczby parametrów i dowolnych przedziałów zmienności.

W celu ustalenia uwagi, nazwijmy przypadek I „mocnym porównywaniem norm", a przypadek II — „słabym porównywaniem norm".

W praktyce występuje zarówno potrzeba słabego porównywania norm, jak i mocnego. Np., jeżeli chcemy zamówić jakiś wyrób za granicą dla konkretnego przeznaczenia w kraju, to wystarczy słabe porównanie norm, aby znaleźć zagraniczny odpowiednik wyrobu, jaki jest nam potrzebny. Jeżeli natomiast chcemy wyjść z jakimś wyrobem na rynki zagraniczne i przekonać kontrahentów o tym, że jest on lepszy od innych wyrobów podobnych — niezbędne stanie się porównanie „mocne", które wykaże, jakie walory posiada nasz wyrób w stosunku do wyrobów konkurencyjnych. Na co dzień w obrocie krajowym wystarcza korzystanie z porównania słabego.

Dodajmy, że niewielka modyfikacja formuły I lub II wystarcza, aby można było ograniczyć lub zmienić warunki porównywalności.

Np. bez trudu możemy z formuły II uzyskać formułę porównywalności typu: X jest zamiennikiem Y, jeżeli przynajmniej jeden (dwa trzy, ...,) parametry spełniają podane warunki.

Ze względu na łatwość dokonywania takich przekształceń, nie będziemy się nad nimi zatrzymywać. Chcielibyśmy jedynie podkreślić, że formuły te umożliwiają algorytmizację porównywania norm opisanych za pomocą parametrów dla dowolnego kryterium.

3. METODA TRADYCYJNA

Tradycyjna metoda porównywania norm polegała na tworzeniu tzw. tablic przejścia. Aby skonstruować taką tablicę dla wybranej grupy wyrobów, należało:

1. Ręcznie zestawić listy parametrów dla wszystkich wyrobów we wszystkich krajach

2. Wybrawszy listę jednego z krajów za podstawową (np. normy polskie), należało kolejno dla każdego wyrobu przeglądać listę parametrów, wyszukując i odnotowując te parametry, które były zgodne

3. Czynność tę powtarzać dla pełnej listy norm każdego kraju. Czynności te są bardzo pracochłonne i żmudne; łatwo przy tym o popełnienie błędu.

W cytowanym przykładzie porównywania norm dla polskiego przemysłu obrabiarkowego w związku z kooperacją z NRD, opracowywanie zamienników norm (tj. norm odpowiadających sobie ze względu na: skład chemiczny, obróbkę mechaniczną, obróbkę termiczną) zajęło sposobem tradycyjnym 10 miesięcy.

Wykorzystanie elektronicznych maszyn cyfrowych pozwala skrócić czas opracowania do kilkunastu czy nawet kilku godzin, zależnie od klasy maszyny i wielkości problemu. Taka oszczędność czasu daje niewątpliwe efekty gospodarcze w postaci możliwości znacznie szybszego rozpoczęcia produkcji. Dlatego przy analizie efektywności zastosowań maszyn cyfrowych do porównywania norm trzeba brać pod uwagę nie tylko koszt bezpośredni metody tradycyjnej i nowej, ale i korzyści uzyskiwane dzięki skróceniu czasu porównywania.

Duża liczba norm i duża liczba porównywanych parametrów przy znajdowaniu zamienników wymaga wiele czasu. Ważnym momentem w mechanizacji procesu jest fakt, że z biegiem czasu część norm ulega dezaktualizacji i należy opracowywać nowe normy, uaktualniając dawne tablice przejścia. Zachodzi więc potrzeba opracowywania nie tylko dużej ilości informacji, ale i informacji stale aktualizowanej. Przemawia to również za celowością zastosowania nowoczesnej techniki obliczeniowej.

Chodzi więc nie tylko o jednorazowe porównywanie, ale i o zaprojektowanie systemu gromadzenia i wyszukiwania informacji.

W niektórych polskich ośrodkach obliczeniowych bądź też w firmach zagranicznych opracowano projekty takich automatycznych systemów. Dla maszyn dostępnych w Polsce utworzono następujące systemy:

Opracowane w Polsce

1. IGA — maszyna IBM-1440 — (na zlecenie CIINTE)
2. IMBI — maszyna ZAM-21 — przez Instytut Maszyn Matematycznych
3. IBIS — maszyna ZAM-41
4. KWOC — maszyna ZAM-41 — CIINTE, IMM
5. SAWI — maszyna MIŃSK-22 — PRETO.

Opracowane za granicą

1. ARDIS — maszyna MIŃSK-22 (Czechosłowacja)
2. APIS — maszyna ELLIOTT 803B
3. SLIC — maszyna ICT-1904
4. PIND — „ „
5. KWIC — „ „

Nie wszystkie jednak z tych systemów zostały uruchomione i zastosowane w Polsce. Prace eksperymentalne trwają.

4. METODA AUTOMATYCZNEGO PORÓWNYWANIA NORM

Opierając się na przedstawionych formułach porównywania można przystąpić do opracowania założeń systemu automatycznego porównywania norm. Założenia te wymagają sprecyzowania następujących elementów:

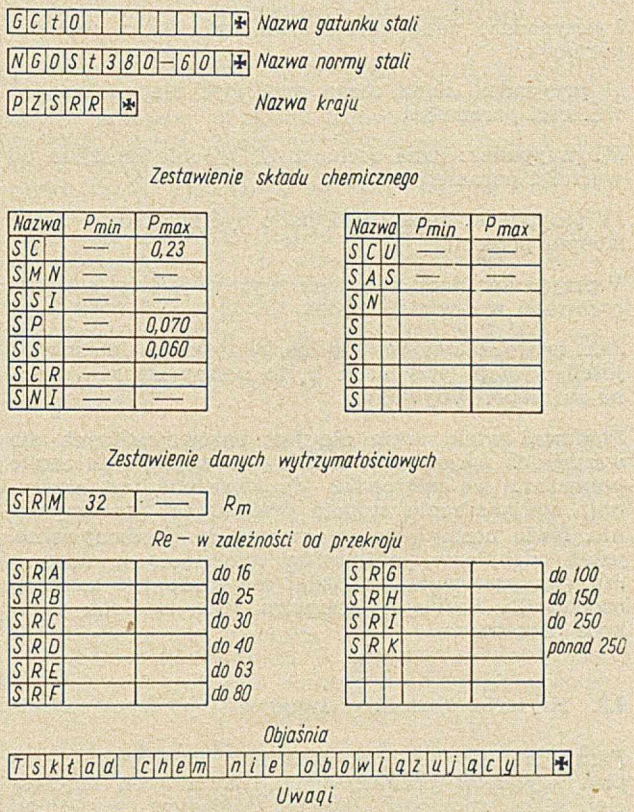
- informacji podstawowej, stałej dla danego zbioru norm, które chcemy porównywać
- zbioru parametrów
- kryteriów porównywania.

4.1. Informacja podstawowa

Aby móc wykorzystać elektroniczną technikę obliczeniową, należy ujednolicić zapis informacji wejściowej. Informację podstawową zapisuje się na przystosowanych do tego kartach danych, które wykorzystane zostaną do tworzenia „maszynowej postaci informacji". Karta taka może zawierać dwa rodzaje informacji:

- informacje identyfikujące (słowniki i indeksy)
- liczbowe wartości poszczególnych parametrów normy.

Wzór przykładowy karty danych opracowanej do porównywania norm stali węglowych podano na rysunku.



Zestawienie danych normowych gatunku stali

We wstępnym etapie opracowania zachodzi potrzeba sporządzania indeksów nazw:

- gatunków porównywanej grupy wyrobów
- norm
- składników wyrobu (parametrów)

oraz opracowania

- formularzy danych o gatunkach grupy wyrobów
- formularzy zapytań.

Przy sporządzaniu indeksów nazw wygodnie jest wykorzystywać istniejące indeksy tam, gdzie mają one postać sformalizowaną numeryczną bądź alfanumeryczną, opatrując je na początku identyfikatorem literowym lub numerycznym.

Np. w wykonanym opracowaniu porównywania gatunków stali węglowych, w oparciu o które dokonano weryfikacji proponowanej metody, przyjęto:

a) dla indeksu nazw gatunków stali symbol składający się z 10 dowolnych znaków dalekopisowych w kodzie M2, opatrzonych na początku znakiem *G*, a na końcu znakiem „krzyża maltańskiego". Litera *G* spełnia tu rolę identyfikatora, który mówi, że symbol występujący po tej literze do znaku krzyża maltańskiego jest symbolem nazwy gatunku stali węglowej,

b) dla indeksu nazwy norm zastosowano jako identyfikator literę *N*, po której następuje symbol 10-znakowy, identyfikator, znak krzyża maltańskiego,

c) dla indeksu nazw państw zastosowano literę *P*, po której występuje 5 znaków i znak krzyża maltańskiego,

d) dla nazw parametrów stali identyfikator *S*, po którym występują 2 znaki i znak krzyża maltańskiego.

4.2. Parametry

Charakterystyka wyrobu za pomocą wektora parametrów może być następująca:

I przypadek: dana cecha musi być równa wartości parametru

II przypadek: dana cecha musi być nie większa od wartości parametru

III przypadek: dana cecha musi być nie mniejsza od wartości parametru.

IV przypadek: dana cecha musi być zawarta w określonym przedziale

V przypadek: dana cecha nie występuje (nie jest precyzowana w normie).

Jak łatwo zauważyć, dla porównywania „mocnego", jeżeli wystąpi przypadek V, to normy zostaną uznane za nieporównywalne.

Ponieważ wiele norm nie jest porównywalnych dla wszystkich parametrów (choćby ze względu na częste pojawianie się przypadku V), konieczne jest zazwyczaj wprowadzenie stopnia dokładności porównywania. Dwie normy uznane zostaną za porównywalne, jeżeli odchylenie parametrów mieści się w dopuszczalnym przedziale. Przedział ten może być ustalany osobno dla każdego parametru jako ustalona wielkość procentowa.

4.3. Kryteria. Formularz zapytań

Formularz zapytań jest nośnikiem służącym do zapisu kryteriów porównywania. Zawiera on najczęściej kilka rodzajów zapytań (rodzajów kryteriów) np.:

1. Wyszukaj gatunki stali, których wektory parametrów normy odpowiadają wzorcowemu wektorowi parametrów.

Kryterium to znajduje zastosowanie, gdy np. projektowana konstrukcja wymaga zastosowania stali o pewnych cechach. Celem porównania jest znalezienie odpowiedniego gatunku stali, aby kierując się dalej innymi kryteriami, np. ekonomicznymi — wybrać najodpowiedniejszy.

2. Wyszukaj gatunki stali, których wektory parametrów odpowiadają wektorowi parametrów wybranego gatunku porównywanego. Kryterium to jest właściwe wówczas, gdy np. nie możemy zakupić gatunku, który do tej pory wykorzystywaliśmy i chcemy znaleźć inny gatunek stali, dostępny na rynku, odpowiadający parametrom gatunku niedostępnego.

3. Kryterium łączące zapytania (1) i (2): wyszukaj gatunki, stanowiące odpowiedniki dla każdego gatunku wchodzącego w skład danej normy, opisanej wektorem parametrów zadanych (jak w p. 1). Zapytania te odpowiadają różnym potrzebom, dla których sporządza się porównanie.

Ponadto niezbędne jest określenie rodzaju porównywania (mocne, słabe) oraz podanie, jakich parametrów porównanie dotyczy — wszystkich czy tylko niektórych, a jeżeli niektórych, to jakich. Dodatkowo trzeba w programie uwzględnić postulat co do traktowania parametrów nie precyzowanych w jednej normie, a precyzowanych w innej porównywanej.

5. DZIAŁANIE SYSTEMU AUTOMATYCZNEGO PORÓWNYWANIA przy UŻYCIU ELEKTRONICZNEJ MASZYNY CYFROWEJ

System porównywania norm przy użyciu komputerów składa się z trzech rodzajów programów:

- programu wczytującego

Zadaniem programu jest wczytanie informacji o poszczególnych gatunkach grupy wyrobów (indeks nazwy, indeks normy, państwa itd.), liczbie komórek objaśnień (tekstu), indeksie nazwy cechy, wartości parametrów. Dane te są umieszczone na taśmie magnetycznej. Wynikiem działania tego programu jest założenie kartoteki norm;

- programów wczytujących pytania, realizujących proces porównywania i zapisujących wyniki porównania na taśmie magnetycznej, według algorytmu opisanego wyżej (pkt. 2);

Zadaniem tej grupy programów jest zapis kryterium porównawczego w pamięci maszyny, zorganizowanie przeglądu informacji podstawowej o gatunkach stali wraz z porównaniem i zapis pozytywnych wyników porównań w formie przystosowanej do drukowania. Proces porównania realizowany jest dla każdego zapytania oddzielnie. Pozwala to na znaczne uproszczenie programów, jak również na rozszerzenie ilości i stopnia skomplikowania kryteriów porównywania;

- programów drukujących tablice przejścia

W zestaw tych programów wchodzą programy sortujące, których zadaniem jest przygotowanie przejrzystej i uporządkowanej postaci informacji oraz do drukowania sam program drukujący. Zależnie od potrzeby można drukować pełne informacje o danym gatunku i gatunkach uznanych za porównywalne lub też tylko interesujące fragmenty tych informacji.

6. WERYFIKACJA METODY na PRZYKŁADZIE PORÓWNYWANIA NORM WĘGLOWYCH STALI PRODUKCYJNYCH

Weryfikację metody porównywania norm dokonano na przykładzie porównywania norm polskich i zagranicznych dla gatunków stali węglowych za pomocą maszyny cyfrowej. W pierwszym etapie wybrano 5 krajów, dla których porównano zgodność norm w oparciu o następujące kryteria:

- zgodność własności mechanicznych
- zgodność własności chemicznych
- zgodność własności mechanicznych i składu chemicznego.

Specjaliści hutnicy opracowali zestawienia gatunków stali krajowych i zagranicznych (ZSRR, CSSR, NRD, NRF) w ustalonej formie, jednoznacznie określającej parametry gatunków stali.

Projektanci systemów EPD ustalili wzór karty danych, jak na rysunku, na które to karty przeniesiono informacje, stanowiące dokumentację pierwotną. Opracowano dwie wersje programów NORM-1 i NORM-2 na maszynie MIŃSK 22 w języku MAT. Pakiet programów NORM-1 zawiera następujące programy:

1. Program zakładania biblioteki norm na taśmie magnetycznej. Ekstremalne wartości parametrów nie mogą przekraczać 400,00. Program wykorzystuje jedną jednostkę taśmy magnetycznej, co pozwala zapisać na niej ok. 3000 norm.

2. Program wybierający normy ze zbioru według ustalonych kryteriów i porządkujący zbiór do drukowania.

3. Program drukujący wyniki na drukarce wierszowej MIŃSK 22. Drukuje się nazwę gatunku stali, symbol normy, nazwę kraju pochodzenia normy, parametry w trzech rodzajach: maksimum składnika, minimum składnika, przedział składników oraz teksty objaśniające.

Pakiet programów NORM-2 różni się tym od pakietu NORM-1, że w porównaniu z NORM-1 zniesiono ograniczenie dotyczące ekstremalnej wielkości parametrów oraz wprowadzono drukowanie tekstu kryterium porównawczego. Liczba porównywanych parametrów nie może przekraczać 50.

Dalszym etapem prac powinna być „uniwersalizacja" programu i zwiększenie liczby norm, które można porównywać. Chodzi tu przede wszystkim o opracowanie programu działającego z większą liczbą jednostek taśmy magnetycznej, aczkolwiek dla wielu przypadków praktycznych jedna jednostka może być wystarczająca.

Programy zostały opracowane w Pracowni ETO COBN dla maszyny MIŃSK 22 i uruchomione w ZETO Katowice. Prace nad programem rozpoczęto w grudniu 1968 roku. Wstępne wyniki uzyskano w marcu 1969 r. Warto wspomnieć, że w czasie międzynarodowego Sympozjum na Temat Zastosowania Matematyki w Normalizacji (20—23 maja 1969 r.) po referacie na temat automatycznego porównywania norm za pomocą maszyn cyfrowych odbył się pokaz pracy systemu przy wykorzystaniu maszyny MIŃSK 22 w Centrum Obliczeniowym Komisji Planowania przy Radzie Ministrów.

Każdy z uczestników Sympozjum otrzymał blankiet, na którym mógł napisać kryterium porównawcze. Po wyperforowaniu zapytania na dalekopisie i wprowadzeniu do komputera w bardzo krótkim czasie zostały wydrukowane wyniki porównania. Czas pełnego cyklu obliczeniowego, liczac od momentu perforacji, do drukowania wyniku przez komputer — wynosił około 4 minuty. Wyniki były za każdym razem poprawne. Nawet w przypadku, gdy jeden z gości zagranicznych postawił specjalnie kryterium nierealne, maszyna i tym razem nie zawiodła, dając odpowiedź, że takiego gatunku stali nie ma.

7. WNIOSKI

7.1. Warunki przy wykonywaniu porównań norm

Dotychczasowe doświadczenia w eksploatacji systemów NORM-1 i NORM-2 dowodzą, że stosowanie elektronicznych maszyn cyfrowych do porównywania norm jest celowe i opłacalne. Niemniej trzeba stwierdzić, że techniczne możliwości maszyn dostępnych w Polsce są zbyt małe, aby można było wprowadzić na dużą skalę prace w zakresie porównywania norm. Przede wszystkim chodzi o to, że aktualny system pozwala na tworzenie kartoteki nie większej niż pojemność jednej jednostki taśmy magnetycznej. Zawęża to dość poważnie zakres zastosowań systemów.

System porównywania norm, jak większość systemów, które w przyszłości znalazłyby zastosowanie w pracach normalizacyjnych, zaliczyć można do klasy systemów gromadzenia i wyszukiwania informacji (*„storage and retrieval systems"*). Jak wiadomo, systemy takie wymagają dużych pamięci o dostępie swobodnym.

Tak więc dopiero maszyny, wyposażone w dyski magnetyczne i to dyski wymienne mogłyby zaspokoić zapotrzebowanie normalizatorów. W Polsce opracowano kilka takich systemów. Jednak tylko system IGA na IBM 1440 wykorzystuje dyski magnetyczne. Pozostałe systemy współpracują z taśmami magnetycznymi. Niemniej biorąc pod uwagę zastrzeżenia dotyczące niższej opłacalności systemów porównywania norm przy wykorzystaniu taśm magnetycznych, trzeba powiedzieć, że są one znacznie efektywniejsze niż dotychczas stosowane tradycyjne metody porównywania. Dotyczy to zarówno czasu, który skracany jest wielokrotnie, jak i kosztu. Koszt, wysoki dla pierwszego porównania, przy szerokim stosowaniu powinien się bardzo poważnie obniżyć dzięki rozłożeniu się kosztu projektu i oprogramowania na wiele przeliczeń.

7.2. Inne systemy do wykonywania porównań

Nie wszędzie ośrodki normalizacyjne mają szybki dostęp do elektronicznych maszyn cyfrowych. Z tego względu do celów bieżącego porównywania norm dla doraźnych potrzeb mogą być wykorzystywane stacje maszyn analityczno-liczących, istniejące w każdym województwie. Przemawia za tym również fakt, że kadra normalizacyjna przygotowana do korzystania z ETO jest niewielka. Zaleta proponowanego systemu i metody jest to, że nośniki informacji (karta norm) nadają się do wykorzystania przy użyciu dowolnej techniki mechanicznego przetwarzania danych. Zapis zaprojektowany dla systemu NORM-1 czy NORM-2 nadaje się również do wykorzystania przez maszyny analityczne pracujące w systemie kart dziurkowanych.

Wielokrotne sortowanie na szybkich sorterach pozwoli na wybranie odpowiednich norm i ich porównanie. Jednocześnie kartoteka na kartach dziurkowanych jest łatwiejsza do aktualizacji. Ponadto warto podkreślić, że sortowanie kart za pomocą szybkich sorterów jest efektywniejsze ekonomicznie niż sortowanie na taśmach magnetycznych. Natomiast w przypadku potrzeby szybkiego porównania norm można wykorzystać tę samą kartotekę do obliczeń na komputerze za pomocą programów NORM-1 czy NORM 2, po dokonaniu konwersji informacji z kart dziurkowanych na taśmę papierową.

Z powyższego wynika, że prowadzenie prac badawczych i przygotowywanie pakietu programów składających się na kompleksowy system automatycznego przetwarzania informacji normalizacyjnej trzeba uznać za celowe i niezbędne: jego efekty pojawią się z chwilą, gdy uzyskamy odpowiednie środki w postaci sprzętu liczącego.

W przypadku porównywania norm parametrycznych na inne wyroby zajść może potrzeba adaptacji programu, lecz zagadnienie to nie będzie trudne wobec szczegółowo sformułowanej ogólnej metody porównywania.

Omówione programy NORM-1 i NORM-2 znajdują się w Zakładzie Obliczeniowym Katowice oraz we wszystkich ośrodkach wyposażonych w maszyny MIŃSK 22.

REWOLUCJA 1970

Większość naszych poglądów i wyobrażeń ciągle jeszcze nosi piętno rewolucji, które miały miejsce w okresie poprzedzającym epokę przemysłową. W czasach tych zarysowywały się poważne i głębokie różnice bogactw, siły i przywilejów, a mała i wyraźnie wyodrębniona klasa rządząca gnębiła i eksploatowała masy ludowe. Dziś każda rzetelna dyskusja na temat rewolucji w krajach rozwiniętych musi bazować na stanie realnym, jakim jest istnienie społeczeństwa technologicznego, a nie operować pojęciami przystosowanymi do analizy społeczeństwa tradycyjnego. Przez „społeczeństwo technologiczne" rozumiem przemysł komputerowy w jego obecnym stadium rozwoju oraz te przedsiębiorstwa i jednostki, które usiłują czerpać korzyści tak na bazie tego przemysłu, jak i z niego samego.

Główna cecha przemysłu komputerowego nie ogranicza się do zwykłej szybkości zmian; przemysł ten niesie ze sobą — jak mówią jego zwolennicy — twórczą destrukcję. Burzy on nie tylko zwyczaje, wierzenia i instytucje odziedziczone z przeszłości, ale i te, które stworzone były nie dalej niż wczoraj. W przemyśle komputerowym pojęcie takie jak „tradycja", czy kategoria taka jak „przyszłość" nie kryją w sobie żadnego sensu. Rewolucja 1970 oznacza raczej walkę przeciw zmieniającej się szybko teraźniejszości niż przeciwko balastowi przeszłości. Oznacza to również, że zamiast tradycyjnej walki przeciw społecznościom i technologiom wykazującym stagnację i brak wzrostu — mamy do czynienia z protestem przeciwko społeczeństwu technologicznemu, które pozostaje w ciągłym ruchu i które jest w stanie obiecywać wszystko od zlikwidowania nędzy do wyzwolenia się od nieuchronności śmierci.

Opuszczając pierwsze dziesięciolecia naszego przemysłu wchodzimy w lata siedemdziesiąte wyposażeni w największy arsenał siły technologicznej, jaki kiedykolwiek był człowiekowi dostępny. Tej nadchodzącej dekadzie musimy postawić zadanie w postaci zlikwidowania naukowego i technicznego balastu, jaki narósł wokół spraw siły technologicznej. Lata siedemdziesiąte stawiają nas wobec wielkiego intelektualnego zadania, które polegać będzie na przeanalizowaniu wszystkich aspektów spuścizny przemysłu komputerowego. Po przebyciu trudów lat pięćdziesiątych i burz lat sześćdziesiątych znowu znaleźliśmy się u progu. Tym razem zadania, które przed nami stoją, nie należą ani do technicznych, ani do finansowych, ale są bardziej zasadnicze i daleko bardziej krytyczne. Musimy przeanalizować, jaki wpływ wywarły komputery na nasz język, literaturę, sztukę, politykę i pracę. Będzie to retrospekcja i pod tym względem, że pociągnie konieczność wyeksponowania wyborów, jakich dokonaliśmy w imię nauki i technologii. Musimy dojść do jednoznacznej oceny strat i zysków, które były rezultatem naszych posunięć. I w końcu aspekt twórczy naszego zadania polegać będzie na zdobyciu pełnej wiedzy o zakresie możliwości, jakie stwarzają komputery.

Zanim nabierzemy rutyny w rozwiązywaniu problemów lat siedemdziesiątych spróbujmy zidentyfikować pewne błędy popełnione w okresie dwóch poprzednich dekad. Dla wygody usystematyzujmy te błędy według następujących grup: istota korporacji, ludzie, urządzenia i programy.

Chaos w ramach korporacji

Dzisiejsze jednostki gospodarcze zaliczane są przez ich własne kierownictwo do grupy nowoczesnych głównie na podstawie takich faktów, jak zakup i eksploatacja urządzeń komputerowych. A tymczasem rozpowszechnienie się wyspecjalizowanej wiedzy technicznej zmieniło współczesną korporację w oszałamiająco złożoną strukturę oddzielnych departamentów, ośrodków komputerowych, ośrodków szkolenia, grup programistów, oddziałów systemów i procedur, i wielu innych. Jednostki te mają często półautonomiczny status w ramach korporacji, dysponują własnymi budżetami, kierowane są przez własny personel techniczny, tworzą własne standardy i w dużym stopniu uzależnione są od tego, co się dzieje poza korporacją. W ten sposób współczesna, oparta na komputerze kompania staje się czymś na kształt firmy posiadającej akcje, która to firma utrzymuje tylko nominalną kontrolę nad oddziałami zarejestrowanymi pod jej nazwą. Rezultatem tego stanu korporacji jest zjawisko nie notowane w naszym kraju od czasu znachorów — domokrążców. Chodzi o to, że zastosowanie komputera uznane zostało przez kierownictwo korporacji za swojego rodzaju lekarstwo uspokajające. Nawet fakt, że koszty związane z zastosowaniem komputera były w minionych dziesięcioleciach niezmierzalne i w dużym stopniu niepoliczalne, jak również zdecydowane prawdopodobieństwo tego, że koszty związane z zastosowaniem komputerów znacznie przewyższały koszty metod ręcznych, niewielki wywarły wpływ na osłabienie romantycznego związku między komputerem i korporacją.

Ludzie

Błędy popełnione w minionych dekadach, a dotyczące spraw ludzi, są nieubłaganie związane z działalnością techniczną korporacji. Podstawowym błędem była tendencja wprowadzania do statusu zawodowego prac, wymagających nie więcej niż odpowiednio przeszkolonej kadry. W ten sposób zastąpiliśmy zawód specjalizacją.

W żadnym innym przemyśle nie potwierdza się w tak oczywisty sposób tzw. Zasada Piotra, a mianowicie że „na drabinie hierarchii każdy pracownik dąży do wzniesienia się na szczebel, gdzie wymagania zaczynają przerastać jego możliwości". Humor zawarty w tym stwierdzeniu jest uderzająco żałosny. Tempo zmian w przemyśle, zwiększające się możliwości wyboru i wzrost korporacji w szybko rozwijającej się gospodarce doprowadzają do nadawania ludziom roli worków z piaskiem, za pomocą których ma być powstrzymana powódź. Istniejący brak personelu technicznego został uznany za stan faktyczny. Problem szkolenia był po prostu ignorowany, a poważne role powierzało się niewłaściwym ludziom.

Konieczność zmian w przemyśle komputerowym nie wymaga, abyśmy uprawiali sztukę chodzenia, zanim nie nauczymy się pełzać, lecz by ci, którzy potrafią pełzać, zaczęli biegać. W ten sposób każdy działa obecnie w otoczeniu trzeciej generacji, przy czym jeżeli w ogóle wykazuje znajomość technik drugiej generacji, to na pewno niewielką. Co więcej, od techników drugiej generacji żądano, by nie rozciągali swoich umiejętności technicznych na trzecią generację — i dążyli do zupełnie nowych rozwiązań. Tą drogą każda nadchodząca generacja niszczy mistrzów generacji poprzedniej, a na ich miejsce wprowadza neofitów generacji bieżącej.

Wśród tego deficytowego stanu i niewłaściwego wykorzystania siły ludzkiej, lata sześćdziesiąte stworzyły zamkniętą organizację technicznej społeczności. Praktykantów ledwie tolerowano i przedostanie się do omawianej sfery działalności było niezwykle utrudnione. Nie chodziło przy tym o żadne kryteria, według których oceniano by zdolności poszczególnych kandydatów; czynnikiem wystarczającym, który przesądzał o odrzuceniu większości aplikantów był brak doświadczenia. Ale nawet to doświadczenie nie było magicznym kluczem do otwierania sobie drzwi. Starający się o pracę musiał być otrzaskany ze stosowanym żargonem, a często wymagano od niego znajomości posługiwania się nowymi narzędziami.

Jednak zademonstrowanie takich umiejętności ograniczone było z kolei trudnymi, jeżeli nie niemożliwymi do pokonania barierami w postaci wielkiej złożoności tych narzędzi oraz ignorancją, jaką wykazywała w tym względzie większość dyrektorów. I tak zaczęła się gra, w której ani kupujący, ani sprzedający nie wiedzieli, co dostają. Firmami, które odnosiły na tym tle prawdziwe korzyści, były agencje pośrednictwa pracy, żerujące na tej wewnętrznej luce technicznej.
Innym przykładem z cyklu najwyższych ukrytych kosztów przetwarzania danych w minionej dekadzie jest koszt zatrudnienia kompetentnej pomocy przy braku odpowiednich narzędzi umożliwiających mierzenie tych kosztów. Równie zdumiewający jest brak takiej techniki dla mierzenia efektywności angażowanej pomocy, gdyż oznacza to, że jedynym odzwierciedleniem pracy tych ludzi stanie się dodatkowa pozycja w kosztach bieżących przedsiębiorstwa. A zatem, jeżeli chodzi o sprawy ludzi, istota błędu poprzednich dziesięcioleci leży przede wszystkim w braku świadomej polityki zatrudnienia i wykorzystania personelu. Luka między generacjami utworzyła się w taki sposób, że z jednej strony oddzieliła ludzi od kierownictwa, a z drugiej strony ludzi od ich podstawowego narzędzia, jakim jest sam system komputerowy.
Następny zasadniczy błąd, który dotyczy ludzi i który dał się nam we znaki, polega na wytwarzaniu uczucia bezsilności w sprawach oceny, czy choćby wyrażania protestu przeciwko osiągnięciom technologicznym. Osiągnięcia takie były siłą narzucane społeczeństwu i to w taki sposób, aby wywołać uczucie powszechnego niedosytu. Ludzie wierzyli, że skoro kultura komputerowa zrobiła wielki krok naprzód, muszą oni dołożyć wszelkich starań, by się do niej włączyć i w ten sposób uniknąć piętna niższości. Stąd błąd popełniony przez jedną firmę wiodącą pozostaje już na zawsze wzorcem do naśladowania dla innych firm danej gałęzi. Uprzejmość sztabu technicznego została prawdopodobnie spotęgowana przez fakt, że działa on w otoczeniu wspaniałomyślnie stwarzającym takie warunki, w których ustępstwa są zawsze możliwe, a błędy zawsze ukrywane.

Błędy w dziedzinie urządzeń

Nietrudno podsumować błędy popełnione w przeszłych dekadach w zakresie urządzeń. Są one z jednej strony rezultatem planowanego starzenia się maszyn, a z drugiej efektem chorobliwego „za dużo, za szybko". W innych przemysłach, mniej technicznie zależnych, starzenie się motywowane jest głównie względami ekonomicznymi. Znaczy to, że prawidłowa działalność określonej linii produkcyjnej zależy w dużym stopniu od wymienialności danego produktu. Gdyby na przykład wybudowano samochód o okresie życia trwającym 20 lat, byłoby to klęską dla przemysłu samochodowego i wywarłoby wielki negatywny wpływ na gospodarkę kraju. W przemyśle komputerowym jednakże, starzenie się nie jest oparte na zasadach ekonomicznych, ale raczej na takich przesłankach, jak elegancja i dążenie do większych osiągnięć technologicznych.

W ten sposób imperatywem staje się dla naszych inżynierów tworzenie większych, szybszych i bardziej złożonych urządzeń, ponieważ pozwala im na to ogólny stan uprawianej sztuki. To nowe urządzenie, kryjące w sobie zwiększone możliwości, może być ekonomicznie uzasadnione porównaniem stosunku kosztu na transakcję dla nowego urządzenia z takim stosunkiem dla modelu poprzedniej generacji, chociaż koszt uzyskania oszczędności często te oszczędności przewyższa. I w ten sposób „starzenie się osiągnięć" zmusza do przedwczesnego wyrzucania starych urządzeń na śmietnik. Taki właśnie proces Vance Packard określa w swoim *„The Wastemakers"* starzeniem się pragnień. Wprowadzenie nowego produktu, który zawiera pewne nowe rozwiązania, powoduje, że produkt stary staje się mniej pożądany, mimo że jest on w dalszym ciągu sprawny, biorąc pod uwagę jakość czy wydajność.

Odwołajmy się raz jeszcze do samochodów jako analogii. Okazuje się, że ich producenci koncentrowali kiedyś swe wysiłki na zupełnie niewłaściwych zagadnieniach. Nie wydaje się bowiem potrzebne konstruowanie i oddawanie w ręce użytkownika samochodu rozwijającego szybkość 120 mil/godz. Choć zarówno przepisy drogowe, jak i zdrowy rozsądek, zakazują wykorzystywania takiej zdolności, poważny problem, który nazwać by można „samochód jako maszyna śmierci" był w przeszłości całkowicie ignorowany. I tak właśnie dzieje się z komputerami. Jesteśmy, co prawda, w stanie szybciej przetwarzać, umieszczać więcej danych w szybkich pamięciach o bezpośrednim dostępie, zwiększać komunikatywność człowiek — maszyna, budować systemy teletransmisyjne itd., ale szczególnie ważna sprawa, jak ludzie mają te nowe zdolności wykorzystywać, jest poważnie zaniedbana.

Kłopoty na polu programowania

Ostatnia kategoria błędów popełnianych w ostatnich dziesięcioleciach dotyczy systemów programowych. W przeciwieństwie do omawianych już urządzeń, gdzie odnotować można „zbyt wiele", w przypadku systemów programowych tradycyjnie odnotowujemy „za mało". Rozumiem przez to za mało dyscypliny oraz za mało narzędzi i środków poza samym systemem komputerowym. Sztuka programowania skierowana była na osiąganie pożądanych rezultatów każdą możliwą drogą, nie wyłączając czarów, palenia kadzideł, a niekiedy ludzkiego poświęcenia. Sztuka ta wymagała uczenia się nowych języków, co prowadziło w efekcie do osiągania przy tworzeniu nowych systemów takiej samej masy trudności, jaka przypisana była do urządzeń starych. Niewiarygodne koszty, jakie narastały w trakcie realizacji systemów w zeszłych dziesięcioleciach, są podsumowaniem błędów obejmujących korporacje, ludzi i urządzenia. Całość ta jest zatem sumą poszczególnych części i przedstawia dokładną definicję luki technologicznej, wobec której stajemy u progu lat siedemdziesiątych.

Patrząc w przyszłość

Cele poszczególnych grup społeczności technologicznej na lata siedemdziesiąte są stosunkowo rozsądne. Korporacje dążą do objęcia większą kontrolą swoich operacji w zakresie przetwarzania danych, przy jednoczesnym obniżeniu kosztów związanych z ich zapotrzebowaniem w tej dziedzinie. Ludzie poszukują dróg, które doprowadziłyby ich do zdobycia i utrzymania pozycji fachowców. Chcą oni w pełni władać osiągnięciami technologicznymi, a nie być niewolnikami narzędzi, które teoretycznie mają być dla nich pomocą. Producenci chcą zaś, by ich produkty bez trudu znajdowały nabywców i, co ważniejsze, by były przez tych nabywców stosowane w sposób przynoszący określone korzyści.

Realizacja tych rozsądnych celów będzie w dużym stopniu zależała od tego, jaki przyjmiemy stosunek do nowych rozwiązań lat siedemdziesiątych. Typowa dla tych nowych rozwiązań jest tendencja w kierunku zwiększającej się specjalizacji. Ma ona wszelkie dane, by stać się albo intensywnym czynnikiem wzrostu, albo ekonomicznej katastrofy. Ogólnie rzecz biorąc w rezultacie tej tendencji na rynku znajdzie się więcej „rzeczy" na sprzedaż, czyli że sprzedawca będzie w stanie umieścić na rynku więcej dóbr i że ten rynek będzie wysoce konkurencyjny. Wynika z tego, że kupujący będzie miał większy wybór dóbr i to prawdopodobnie po niższych cenach. Lecz, co ważniejsze, może to oznaczać, że zacznie się powtarzać sytuacja lat pięćdziesiątych i sześćdziesiątych, a jedyna różnica będzie miała charakter ilościowy.

Wyodrębnienie sprzedaży

Istniejąca luka technologiczna odziedziczona po latach pięćdziesiątych i sześćdziesiątych została podzielona i utrzymana w podziale na dwie wyspecjalizowane dziedziny: hardware i software. Pozostawiona sile swojej bezwładności obecna specja-

lizacja stwarza warunki dla powtórzenia błędów przeszłości, z tym że teraz będą one bezpośrednio obciążać albo hardware, albo software. Obraz jest bardzo podobny do sytuacji, jaka miała miejsce w momencie, kiedy przechodziliśmy od języków maszynowych drugiej generacji do języków trzeciej generacji. Żadna korporacja nie doszła wówczas do wniosku, że przy zastosowaniu nowych języków należało wprowadzić coś nowego w porównaniu z tym, co robiono poprzednio. Grupy programujące w SPS przenosiły tę samą organizację na nowe warunki, dla których używany był COBOL. Zamiast wprowadzania organizacji, która wykorzystywałaby nowe narzędzia, przystosowywano komputer do starej organizacji.

Przy obecnej specjalizacji tego właśnie błędu należy unikać. Potrzebna jest nie reorganizacja ciała, ale raczej reorganizacja umysłu. Lata siedemdziesiąte postawią wymóg dokładnego przeanalizowania zastosowań komputera i urządzeń pomocniczych. Konieczne będzie przekwalifikowanie starych priorytetów i ustanowienie nowych tak, by odpowiadały rozwojowi technologii. Priorytety te muszą pójść w parze z położeniem znacznie większego nacisku na kształcenie i szkolenie, a jednocześnie z większymi wymogami w stosunku do gwarancji doskonałości i sprawności technicznej.

Aby przystosować te nowe tendencje do komputerowego apartheidu, rozprzestrzeniać się zaczną nowe kompanie. Organizacje te będą początkowo działać jako dystrybutorzy urządzeń, pakietów i środków naukowych. W przeciwieństwie do minionych dziesięcioleci, kiedy to producenci sami zajmowali się dystrybucją swoich produktów, nowe kompanie rozprowadzać będą wyroby wielu producentów. W przeszłości grupa producentów składała się przede wszystkim z potężnych, silnych organizacji, co związane było z ogromnym wkładem inwestycyjnym, jakiego wymagała produkcja urządzeń komputerowych. Producenci software'u jednakże są już zupełnie inną kategorią.

Praktyka podziału na hardware i software może wkrótce doprowadzić do tego, że liczba kompanii, z których każda oferuje dziesiątki pakietów software'owych wzrośnie do 1000. Pakiety te reprezentują inwestycje we własne programy, dokonane zarówno przez wielkie jednostki, jak i przez dyktatorskich programistów. Będą się na nie składać zarówno wielkie systemy jak i małe podprogramy. Obejmą one początkowo wszystkie działy gospodarcze z rynku przetwarzania danych, a w końcu wkroczą w dziedzinę techniki i matematyki. Ponieważ koszty początkowe tego rodzaju firm są niezwykle niskie i ponieważ wydaje się, że firmy takie będą w modzie w latach siedemdziesiątych, użytkownicy znajdą jeszcze jedną przeszkodę na drodze osiągania ich zasadniczego celu, jakim jest racjonalne stosowanie komputerów. Bo jeżeli ma się do czynienia z 1000 kompanii, z których każda oferuje 100 urządzeń i 100 pakietów, prawdopodobieństwo popełnienia błędu staje się co najmniej poważne. Na tym tle wybór przy zakupie jednostki centralnej jest sprawą stosunkowo łatwą.

Zakup komputera stanowi tylko część kosztu, a dla niektórych jest to nawet część niewielka. Sum naprawdę dużych wymaga eksploatacja maszyny, tj. ludzie dla jego obsługi, urządzenia pomocnicze i programy. I na tym właśnie polu lata siedemdziesiąte wystawiają użytkowników komputerów na najcięższą ze wszystkich dotychczasowych prób.

Klucza, od którego zależy sukces w dziedzinie przemysłu komputerowego, nie należy szukać ani w urządzeniach, ani w software'rze, lecz w ludziach. W przeciwieństwie do minionych dziesięcioleci widzimy dziś, jak mury wyższych uczelni i uniwersytetów opuszczają ludzie, którzy są z komputerem, jeśli nie w pełni to częściowo, obeznani. Ten właśnie element stanie się podstawą, na której bazować będą prężne kompanie lat siedemdziesiątych. Drogą tworzenia i wykorzystywania nowych środków nauki i szkolenia oraz nowych technik stworzona zostanie kadra prawdziwych fachowców. Dla takiego właśnie celu wykorzystywany będzie sam komputer, przy czym dążyć się będzie nie tylko do wyszkolenia własnych przyszłych mistrzów, ale i do przygotowania kadry fachowców dla innych dziedzin, tylko pośrednio związanych z komputerami.

Główną różnicą między społecznościami technologicznymi lat siedemdziesiątych i minionych dziesięcioleci będzie prawdopodobnie bazowanie na technikach mierzenia wydajności. Zawierać się w tym będzie oczywiście wydajność ludzi, urządzeń i programów.

Zmiana kryteriów oceny

W minionych dziesięcioleciach wyraźnie dała się odczuć niechęć, jeżeli chodzi o zajęcie się sprawami oceny materiału ludzkiego. Ostatnie opracowania testów powszechnych przygotowane przez DPMA spotkały się z dezaprobatą dużej części społeczności technicznej. Faktem pozostaje jednak, że koszt prowadzenia określonych prac w dziedzinie przetwarzania danych, tak dziś jak i jutro, jest w sposób decydujący uzależniony od wydajności jednostek, prowadzących działalność na polu przetwarzania danych. Od chwili, gdy przezwyciężyliśmy nasze socjologiczne obiekcje w stosunku do klasyfikowania istot ludzkich, przestało być sprawą niemożliwą mierzenie zdolności i wydajności poszczególnych jednostek. Istotą oceniania ludzi nie jest bynajmniej podciąganie wszystkich jednostek pod tę samą kategorię, czy też stwarzanie warunków, w których dążyłyby one do tego samego stopnia doskonałości. Sprawa polega raczej na określaniu zdolności jednostki i przydzielaniu jej tego spośród będących w dyspozycji stanowisk, które pozostaje w zgodzie z jej poziomem zdolności.

Podobnie same systemy muszą być oceniane według ich stopnia złożoności. Dopiero wówczas będziemy w stanie dokonywać rozdziału personelu technicznego na projektantów i programistów zgodnie zarówno ze zdolnościami jednostek jak i wymogami systemu. Równie istotnym zagadnieniem wydaje się wynalezienie technik, które pozwoliłyby oceniać wartość różnorodnych produktów ukazujących się na rynku. Techniki takie powinny umożliwić dokonanie oceny produktu nie w warunkach ogólnych, ale w specyficznym realnym otoczeniu, które jest elementem zasadniczym dla użytkownika realizującego wytknięty sobie cel. Bez tego rodzaju metod pomiarów przemysł komputerowy nie zejdzie ze swego obecnego toru rozwoju, gdzie bardzo niewiele miejsca poświęca się sprawie zastosowań teorii przetwarzania.

Wielce niekorzystną sytuacją, w jakiej od dawna znajduje się cała społeczność użytkowników komputerów, jest poważny wpływ, który skupiają w swych rękach producenci hardware'u. Kontrolują oni gospodarkę, narzucają formę danych, zalecają techniki przetwarzania, które powinno się stosować. W ten sposób samo posługiwanie się podstawowym narzędziem naszego zawodu staje się głównym źródłem bezsilności. Jesteśmy bezsilni, aby powstrzymać postęp, bezsilni, aby powstrzymać zmiany i bezsilni, aby wstrzymać się od kupna. Jeżeli mamy przetrwać, lata siedemdziesiąte muszą położyć kres tej podstawowej chorobie naszego komputerowego społeczeństwa, tj. powszechnemu uczuciu bezsilności zamaskowanemu pod szyldem „zużycie", a utrzymującemu się dzięki rosnącym osiągnięciom technicznym.

Nie ulega najmniejszej wątpliwości, że korporacje lat siedemdziesiątych, działające w sferze naszego technologicznego społeczeństwa, kłaść będą jeszcze większy nacisk na sprawy technologicznych osiągnięć. Korporacja połączona będzie kompletną siecią komunikacyjną ze swoją jednostką centralną, co znaczy, że każdy oddział korporacji będzie bezpośrednio uzależniony od wyników pracy komputera. Zastosowane zostaną najdoskonalsze środki łączności, za pomocą których człowiek i maszyny nawiążą trwały, a przy tym efektywny dialog.

Ale zarówno te rozwiązania, jak i wiele innych nie zdołają wyjść poza fazę technologicznych innowacji,

cd. na III str. okł.

cd. ze str. 24

jeśli my sami nie okażemy się skłonni i gotowi podjąć sprawę przemyślenia i przewartościowania naszych celów. Waga, jaką przypisujemy żądaniu, by najpierw określić jak najdokładniej sam problem, a potem poszukać najwłaściwszej drogi do rozwiązania go jest podstawą naszego istnienia. Nie ma dziś wątpliwości co do tego, że przemysł komputerowy prowadzi nieuchronnie do samounicestwienia się. Każda zasadnicza analiza tego stanu rzeczy musi wyjść z przesłanki, że oczywista siła niszczycielska tego przemysłu nie zostanie wstrzymana na drodze akceptowania istniejących wartości i stosowania rozwiązań, stanowiących źródło problemu. Jeżeli korporacje, wchodzące w skład omawianej technologicznej społeczności, poświęcą swe wysiłki właściwemu przeanalizowaniu obecnej sytuacji, mogą one nie tylko w najlepszy sposób przysłużyć się społeczeństwu, ale mogą nawet uratować siebie.

„Datamation", January 1970.
Howard Bromberg — „Revolution — 1970".

opracowała
Ewa Zawisza

RYNEK KOMPUTERÓW w USA

Firma DIEBOLD w swym najnowszym raporcie o stanie komputerów w USA stwierdza, że przy końcu roku 1969 powinno było być zainstalowanych około 70 000 maszyn cyfrowych. W roku 1968 zainstalowano 15 000 nowych komputerów, a w roku 1969 — 14 000. Z danych tych wynika, że tempo przyrostu komputerów w USA osłabło. Być może sytuacja ta jest tylko chwilowa.

Producenci komputerów w Stanach Zjednoczonych przewidują, że sprzedawane obecnie ulepszone modele trzeciej generacji doprowadzą do zwiększonego popytu. Poza tym w ostatnim okresie wzrosła gwałtownie sprzedaż mini-komputerów. I tak w roku 1968 zainstalowano 3000 mini-komputerów, a w roku 1969 już 4500. Wśród producentów mini-komputerów wybija się na czoło firma DIGITAL EQUIPMENT CORPORATION.

W podziale rynku zbytu pomiędzy poszczególne przedsiębiorstwa nadal przoduje firma IBM, wykazując 57,8% zainstalowanych maszyn cyfrowych. Na drugim miejscu stoi wspomniana firma DEC, mająca 7,6% zainstalowanych maszyn w USA. Na ostatnim miejscu w tabeli znajduje się firma XEROX DATA SYSTEMS, która dawniej występowała pod nazwą SCIENTIFIC DATA SYSTEMS. Firma ta produkuje nadal komputery do zastosowań naukowo-technicznych takie, jak SDS seria 900, SIGMA 2,5 i 7.

Firma CONTROL DATA CORPORATION zajmuje piąte miejsce w tabeli. W roku 1972 zakończone zostaną prace nad komputerem CDC 7600, który będzie wtedy najpotężniejszą maszyną cyfrową świata, podobnie jak kilka lat temu była nią CDC 6600.

Jako ciekawostkę można podać, że system oprogramowania dla pracy z podziałem czasu zostanie opracowany przez francusko-angielską firmę "software'ową" SIA. Firma SIA wyrobiła sobie dobrą markę, opracowując system programowania z podziałem czasu dla maszyn CDC serii 6000 zainstalowanych we Francji i Anglii. System ten znany jest pod nazwą OPHELIE II i będzie prawdopodobnie także stosowany w USA. System programowania dla CDC 7600 będzie się nazywał OPHELIE III. Kontrakt pomiędzy firmą CDC a SIA opiewa na sumę 4 mln dolarów.

„New Scientist" 25.XII.1969 r.

Opracował
Aleksander Senkowski

Tablica

Zainstalowane i zamówione komputery w USA

Stan na dzień 30.VI.1969 r.

Nazwa firmy	Zainstalowane		Zamówione
	sztuk	%	sztuk
IBM	35 811	57,8	8 800
DIGITAL EQUIPMENT CORP.	4 707	7,59	900
UNIVAC	4 344	7,00	450
HONEYWELL	3 219	5,16	600
CONTROL DATA CORP.	1 858	3,00	175
BURROUGHS	1 542	2,49	500
RCA CORP.	1 293	2,08	180
GENERAL ELECTRIC	1 230	1,98	300
XEROX DATA SYSTEMS	1 095	1,77	120
Inne	6 978	11,00	5 225
Razem	62 077	100,00	172 000

Uwaga: Tablica nie oddaje właściwych proporcji wielkości poszczególnych firm amerykańskich, ponieważ nie uwzględnia wartości zainstalowanych maszyn.

WIADOMOŚCI PKAPI

Drugie krajowe spotkanie użytkowników EMC ODRA

W dniach od 9 do 11 lutego 1970 r. odbyło się w siedzibie NOT w Krakowie II Krajowe Spotkanie Użytkowników EMC ODRA.

Pierwszy dzień (9.02.) był poświęcony problematyce Sekcji ODRA 1003/1013 — wymiana informacji o opracowanych programach oraz możliwości adoptowania ALGOLU opracowanego w Czechosłowacji dla m. c. ODRA w Polsce.

Drugi dzień (10.02.) był głównie poświęcony EMC ODRA 1304.

W obradach uczestniczyli członkowie Sekcji ODRA 1003/1013 oraz Sekcji ODRA 1204.

W drugim dniu obrad wygłoszone zostały następujące referaty:

1) mgr Marian Snowarski — „Architektura logiczna systemu ODRA 1304",

2) mgr Fajwel Klein — „Software systemu ODRA 1304",

3) mgr Władysław Bara i inż. Jerzy Nawrot — „Monitor ekranowy dla EMC ODRA 1204".

W imieniu Klubu Użytkowników m.c. ICL wystąpił na spotkaniu dyrektor ZETO GDAŃSK mgr Żydowo deklarując ścisłą współpracę z Klubem Użytkowników m.c. ODRA.

Trzeci dzień (11.02) był poświęcony Sekcji ODRA 1204. W dniu tym wygłoszone zostały następujące komunikaty:

- dr inż. Henryk Siemaszko — o pracach Instytutu Energetyki,
- mgr Romuald Matys — o programach administracyjnych wykonywanych na maszynie ODRA 1204 w Centralnym Ośrodku Badań i Rozwoju Techniki Kolejnictwa,

cd. na IV str. okł.

cd. z III str. okł.

● mgr Zbigniew Juszczyk — o pracach Przemysłowego Instytutu Telekomunikacji.

W imieniu Klubu Użytkowników maszyn ODRA 1204 w Hradec Kralove wystąpił mgr František Veis omawiając główne zagadnienia opracowane przy użyciu m.c. ODRA w Czechosłowacji. Na zakończenie swojego wystąpienia mgr F. Veis zadeklarował współpracę w obustronnej wymianie informacji o opracowanych programach.

Po ożywionej dyskusji użytkownicy EMC ODRA sformułowali następujące postulaty i wnioski:

● wystąpić do PRETO o zwiększenie dostaw na rynek krajowy w latach 1970—71 pamięci bębnowej dla maszyn ODRA 1204,

● wystąpić do PRETO o przyśpieszenie dostaw i zaspokojenie potrzeb rynku krajowego w latach 1970—72 w drukarki wierszowe dla maszyn ODRA 1204,

● doprowadzić do zapewnienia dostaw automatów piszących typu Consul (CSRS) zamiast automatów piszących typu OPTIMA 527 dla maszyn ODRA 1204,

● postulować w PRETO, aby w ośrodkach obliczeniowych, w których prowadzi się poza obliczeniami naukowo-technicznymi również prace z przetwarzania danych umożliwić w przyszłości instalację maszyn ODRA 1304,

● wystąpić do PRETO o natychmiastowe ustalenie rozdzielnika na maszyny cyfrowe ODRA 1304 i ODRA 1204 na okres 1970—1972.

Sprecyzowanie terminów przydziałów jest konieczne ze względu na właściwe przygotowanie jednostek organizacyjnych do prawidłowej eksploatacji maszyn cyfrowych w momencie ich otrzymania.

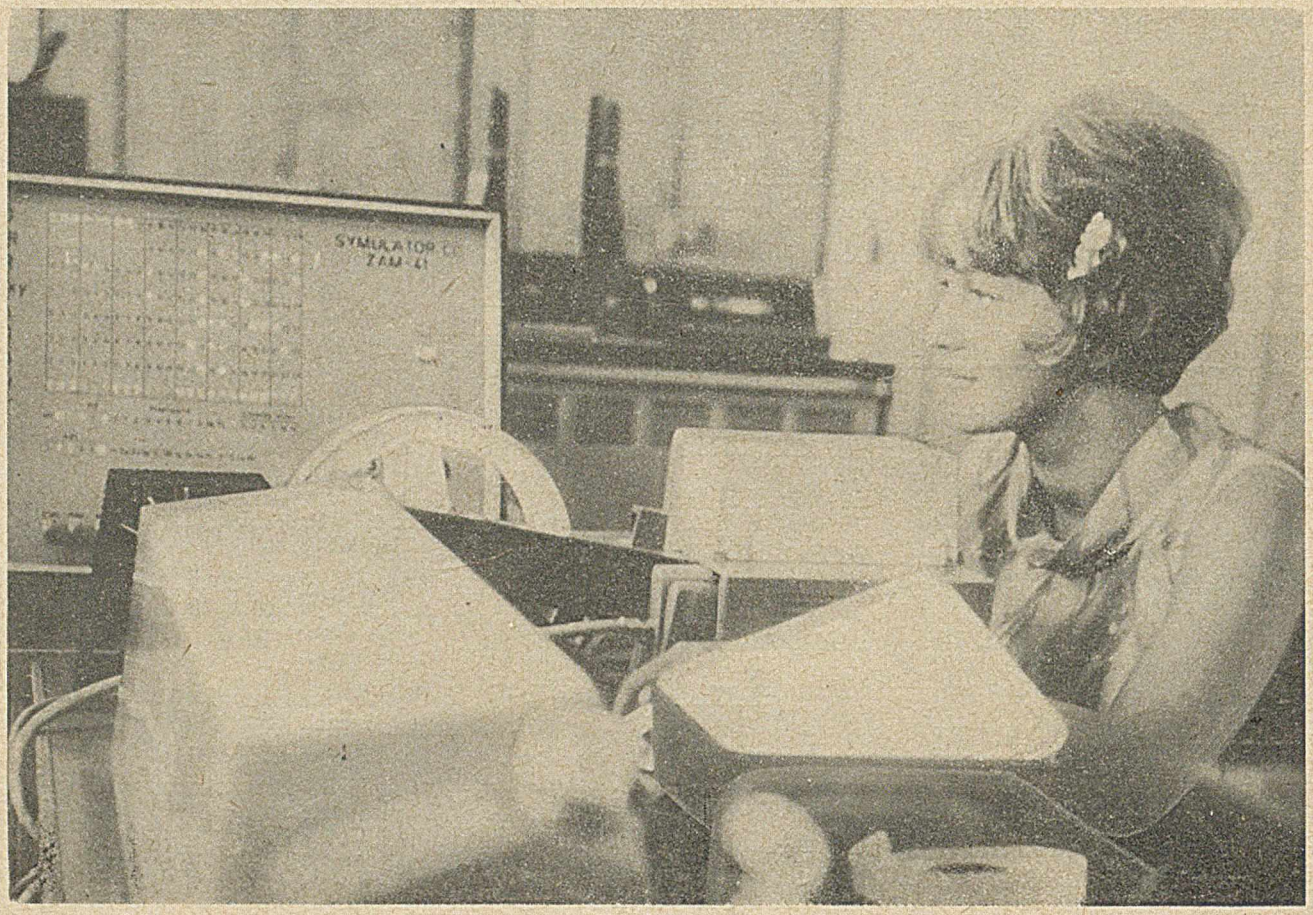
Szczególnie istotną jest sprawa wcześniejszego przeszkolenia personelu technicznego, tak aby w chwili otrzymania maszyny ośrodek otrzymujący miał już przeszkolonych konserwatorów.

● wystąpić do PRETO i Zjednoczenia MERA o spowodowanie produkcji i wprowadzenie do zbytu przez ELWRO modułów urządzeń WE-WY dla zestawów ODRA 1204 i ODRA 1304 w ilościach i konfiguracjach żądanych przez odbiorców.

Na zakończenie spotkania poruszono szereg spraw organizacyjnych.

Należy dodać, że wszyscy dyskutanci podkreślali konieczność wymiany informacji o opracowanych programach oraz o udoskonaleniach technicznych wykonanych w poszczególnych ośrodkach obliczeniowych.

Romuald Matys

Centralny Ośrodek Badań i Rozwoju Techniki Kolejnictwa

Symulator maszyny ZAM-41